Emotion Classification In Software Engineering Texts: A Comparative Analysis of Pre-trained Transformers Language Models

Preprint · January 2024

CITATIONS
0

READS
99

1 author:

Mia Mohammad Imran
Virginia Commonwealth University
10 PUBLICATIONS **18** CITATIONS

SEE PROFILE

Emotion Classification In Software Engineering Texts: A Comparative Analysis of Pre-trained Transformers Language Models

Mia Mohammad Imran
Virginia Commonwealth University
Richmond, Virginia, USA
imranm3@vcu.edu

ABSTRACT

Emotion recognition in software engineering texts is critical for understanding developer expressions and improving collaboration. This paper presents a comparative analysis of state-of-the-art Pre-trained Language Models (PTMs) for fine-grained emotion classification on two benchmark datasets from GitHub and Stack Overflow. We evaluate six transformer models - BERT, RoBERTa, ALBERT, DeBERTa, CodeBERT and GraphCodeBERT against the current best-performing tool SEntiMoji. Our analysis reveals consistent improvements ranging from 1.17% to 16.79% in terms of macro-averaged and micro-averaged F1 scores, with general domain models outperforming specialized ones. To further enhance PTMs, we incorporate polarity features in attention layer during training, demonstrating additional average gains of 1.0% to 10.23% over baseline PTMs approaches. Our work provides strong evidence for the advancements afforded by PTMs in recognizing nuanced emotions like Anger, Love, Fear, Joy, Sadness, and Surprise in software engineering contexts. Through comprehensive benchmarking and error analysis, we also outline scope for improvements to address contextual gaps.

ACM Reference Format:
Mia Mohammad Imran. 2024. Emotion Classification In Software Engineering Texts: A Comparative Analysis of Pre-trained Transformers Language Models. In *Proceedings of 3rd International Workshop on Natural Language-based Software Engineering (NLBSE 2024).* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 INTRODUCTION

The exploration of emotion classification in software engineering (SE) texts has garnered considerable attention in recent years [1, 2], leading to the development of various emotion classification tools by researchers. Notable among these are ESEM-E by Murgia et al. [3], EMTk by Calefato et al. [4], and SEntiMoji by Chen et al. [5]. Chen et al.'s comparative study, utilizing emotion datasets extracted from Stack Overflow and JIRA [6, 7], affirmed the superior performance of SEntiMoji compared to EMTk and ESEM-E, solidifying its prominence in the domain. However, compared to open-domain, Pre-Trained Language Models (PTMs) rarely have been used in software engineering text for emotion classification.

A recent study by Li et al. [8] introduced the use of BERT [9] as part of a two-stage framework for ambiguity detection, surpassing the performance of SEntiMoji in a GitHub emotion dataset curated by Imran et al. [10]. Another recent study by Bleyl et al. [11] found that BERT outperforms EMTk on Stack Overflow dataset curated by Novielli et al. [7]. This shift towards leveraging PTMs prompts a critical inquiry into the overall efficacy of such models in the nuanced task of emotion classification within software engineering contexts. Notably, Pre-trained Transformer Language Models like BERT [9] and RoBERTa [12] have already demonstrated state-of-the-art results across various software engineering domains, including code completion, code review, bug localization, sentiment analysis, and toxicity detection [13–19].

In light of these developments, this paper embarks on a comprehensive exploration of Pre-trained Transformers Language Models in emotion classification for software engineering texts. Through rigorous evaluation and comparison, we aim to shed light on the strengths, limitations, and potential implications of leveraging PTMs in this nuanced domain.

Our study utilizes two distinct datasets sourced from GitHub by Imran et al. [10] and Stack Overflow by Novielli et al. [7], encompassing a diverse range of emotions expressed in software engineering contexts. We assess the effectiveness of six state-of-the-art PTMs, including four general domain models (BERT, RoBERTa, ALBERT and DeBERTa) and two specifically tailored for software engineering (CodeBERT and GraphCodeBERT). This evaluation is conducted against the benchmark set by the SEntiMoji model. EMTk and ESEM-E are excluded from this comparative analysis, aligning with findings reported by Chen et al. [5] indicating the superior performance of SEntiMoji.

Our investigation is guided by two primary research questions, each addressing specific facets of PTMs for emotion classification:

RQ1: How accurately can Pre-Trained Language Models classify emotions in software engineering texts compared to state-of-the-art model?

To address this question, we evaluate the performance of the six aforementioned state-of-the-art PTMs against the SEntiMoji model. Utilizing two distinct datasets, the GitHub dataset by Imran et al.[10] and the Stack Overflow dataset by Novielli et al.[7], both annotated using the comprehensive model by Shaver et al. [20], we scrutinize the models' efficacy in capturing the nuances of emotions expressed in software engineering contexts. Our results suggest that the PTMs generally outperform state-of-the-art, average f1-score improvement ranges from 1.17% to 16.79%.

NLBSE 2024, April 2024, Lisbon, Portugal

ACM ISBN 978-x-xxxx-xxxx-x/YY/MM…$15.00
https://doi.org/10.1145/nnnnnnn.nnnnnnn

RQ2: Can integrating polarity features in the training improve Pre-Trained Language Models' ability for emotion classification?
Building upon observations by Chen et al.[5] and Imran et al.[10] regarding the impact of sentiment polarity on emotion classification errors, we focus on the potential benefits of incorporating polarity features during transformer model training. Specifically, we examine the extent to which positive and negative polarity features enhance the contextual understanding of PTMs for emotion classification in software engineering texts. We incorporate polarity features in attention layer of the PTMs. Our results suggest that they, indeed, improve the baseline PTMs further 1.0% to 10.23% in average f1-score metric.

The main contributions of this work are as follows:

- To our best knowledge, this is the first study which leverages various PTMs for a large-scale comparative study against existing state-of-the-art tool for emotion classification in software engineering text.
- This is the first study which leverages polarity features to enhance PTMs for emotion classification in software engineering text.

We release the source code at URL: https://anonymous.4open.science/status/SE_Emotion_PTM-3589.

2 EXPERIMENT SETUP

2.1 State-of-the-art Models

SEntiMoji: SEntiMoji is a deep learning-based model and pre-trained on GitHub data. The model is proposed by Chen et al. [5] and built on top of the DeepMoji [21] model. This flexible model can identify different emotion categorization schemes as well as sentiment classification.

2.2 Compared PTMs

We compare 6 encoder-based PTMs: BERT, RoBERTa, DeBERTa, ALBERT, CodeBERT and GraphCodeBERT. Previous research shows that BERT, RoBERTa and ALBERT work well in SE affect analysis [17, 22], while DeBERTa shows promising results in sentiment analysis [23]. In addition to these four models, we compare against 2 domain-specific models that are widely used and proposed by Microsoft: CodeBERT and GraphCodeBERT – which showed promising results in various SE tasks [24, 25]. The models are explained below.

BERT [9]: Introduced by Google in 2018, BERT, pre-trained on a diverse corpus, including Wikipedia and Book Corpus, revolutionized natural language processing NLP with its bidirectional transformer architecture. This design allows BERT to capture context from both left and right contexts of a given word, making it highly effective for a diverse range of NLP tasks.

RoBERTa [12]: Developed by Meta AI, RoBERTa represents a refinement in transformer-based models. By modifying key hyperparameters and leveraging an extensive training dataset, RoBERTa enhances its performance on benchmark NLP tasks, showcasing improved language representation understanding.

DeBERTa [26]: An evolution of BERT and developed by Microsoft Research, DeBERTa focuses on enhancing the decoding process in language understanding tasks. It incorporates directional masks during training to capture bidirectional dependencies effectively, showcasing prowess in tasks requiring sequential reasoning.

ALBERT [27]: Designed by Google Research, ALBERT introduces efficiency improvements to the BERT architecture without compromising representational power. This is achieved through parameter reduction techniques during training, showcasing resource-efficient yet highly effective language representations.

CodeBERT [28]: Tailored for programming languages and code understanding, CodeBERT, developed by Microsoft Research, is pre-trained on a large-scale dataset of programming tasks. This specialization allows CodeBERT to excel in source code-related tasks such as code summarization and variable naming.

GraphCodeBERT [29]: Developed by Microsoft Research, GraphCodeBERT is a transformer-based model designed for comprehending programming languages and code. Pre-trained on a vast dataset of programming tasks, it effectively captures the complex structures and semantics of code, serving as a proficient solution for source code-related tasks.

We use the publicly available versions of each model in Hugging Face [30]. The model versions are shown in Table 1.

Table 1: Used model versions on Hugging Face

Model	Version
BERT	bert-base-uncased
RoBERTa	roberta-base
ALBERT	albert-base-v2
DeBERTa	microsoft/deberta-v3-base
CodebERT	microsoft/codebert-base
GraphCodeBERT	microsoft/graphcodebert-base

2.3 Datasets

We conduct experiment using two existing SE-emotion datasets. Both datasets are annotated using Shaver's emotion model [20]. Shaver's emotion model has six basic emotion categories: Anger, Love, Fear, Joy, Sadness, and Surprise. The datasets are:

GitHub Emotion Dataset. Imran et al. [10] curated a multi-label emotion dataset that is crawled from GitHub. The dataset consists of 2000 data points. The dataset contains 340 (17.0%) Anger, 220 (11.0%) Love, 198 (9.9%) Fear, 422 (21.1%) Joy, 274 (13.7%) Sadness, and 328 (16.4%) Surprise comments. The rest are neutral comments.

Stack Overflow Emotion Dataset. Noveilli et al. [7] curated a multi-label emotion dataset from Stack Overflow Discussion. The dataset consists of 4800 data points. The dataset contains 882 (18%) Anger, 1220 (25%) Love, 106 (2%) Fear, 491 (10%) Joy, 230 (5%) Sadness, and 45 (1%) Surprise. The rest are neutral comments.

For both datasets, we do simple text-preprocessing by lowercasing and removing null characters.

2.4 Metrics

We evaluate using F1-score – which is commmonly used for this evaluation [5, 10]. F1-score is the harmonic mean of Precision and Recall: $F1\text{-}score = 2 * \frac{Precision * Recall}{Precision + Recall}$. Precision measures the ratio of true positives to all positive predictions, while Recall calculates the ratio of true positives to all actual positives. Additionally,

Table 2: Evaluation of PTMs on the Emotions Dataset (F1-score).

Dataset	Model	Anger	Love	Fear	Joy	Sadness	Surprise	Micro Avg.	Macro Avg.
GitHub	SEntiMoji	0.460	0.642	0.377	0.556	0.629	0.458	0.530	0.521
	BERT	0.426	0.731	0.545	0.597	0.609	**0.639**	0.585	0.591
	(+/-)	-7.49%	+13.83%	+44.55%	+7.29%	-3.26%	+39.49%	+10.37%	+13.56%
	RoBERTa	0.517	**0.774**	**0.561**	0.521	**0.653**	0.511	0.575	0.590
	(+/-)	+12.44%	+20.60%	+48.66%	-6.37%	+3.85%	+11.55%	+8.45%	+13.28%
	ALBERT	0.446	0.753	0.357	0.447	0.631	0.602	0.538	0.539
	(+/-)	-2.98%	+17.29%	-5.36%	-19.61%	+0.23%	+31.30%	+1.52%	+3.61%
	DeBERTa	**0.578**	0.736	0.476	**0.605**	0.642	0.611	**0.610**	**0.608**
	(+/-)	+25.72%	+14.59%	+26.19%	+8.69%	+1.95%	+33.33%	+15.07%	+16.79%
	CodeBERT	0.446	0.653	0.548	0.518	0.591	0.574	0.545	0.555
	(+/-)	-3.01%	+1.73%	+45.21%	-6.86%	-6.09%	+25.19%	+2.95%	+6.62%
	GraphCodeBERT	0.476	0.632	0.507	0.552	0.551	0.578	0.549	0.549
	(+/-)	+3.52%	-1.62%	+34.27%	-0.74%	-12.43%	+26.14%	+3.60%	+5.53%
Stack Overflow	SEntiMoji	0.759	0.819	0.429	0.435	0.556	0.182	0.714	0.530
	BERT	0.769	0.851	0.545	0.597	**0.600**	0.167	0.754	0.588
	(+/-)	+1.40%	+3.95%	+27.27%	+37.05%	+8.00%	-8.33%	+5.62%	+11.02%
	RoBERTa	0.786	**0.872**	0.581	0.591	**0.600**	0.1667	**0.758**	0.599
	(+/-)	+3.60%	+6.49%	+35.48%	+35.77%	+8.00%	-8.33%	+6.23%	+13.13%
	ALBERT	0.762	0.845	0.579	**0.640**	0.545	0.133	0.747	0.584
	(+/-)	+0.40%	+3.13%	+35.09%	+47.00%	-1.82%	-26.67%	+4.65%	+10.23%
	DeBERTa	**0.777**	0.860	**0.591**	0.604	0.598	**0.211**	0.756	**0.607**
	(+/-)	+2.49%	+5.04%	+37.88%	+38.68%	+7.59%	+15.79%	+5.90%	+14.52%
	CodeBERT	0.772	0.854	0.556	0.519	0.537	0.167	0.728	0.567
	(+/-)	+1.79%	+4.29%	+29.63%	+19.17%	-3.41%	-8.33%	+2.04%	+7.08%
	GraphCodeBERT	0.727	0.836	0.514	0.559	0.521	0.154	0.722	0.552
	(+/-)	-4.13%	+2.06%	+20.00%	+28.32%	-6.30%	-15.38%	+1.17%	+4.14%

micro-averaged F1 considers the overall performance by aggregating individual class scores, treating each instance equally, whereas macro-averaged F1 computes the average across emotion classes, providing equal weight to each class regardless of their size.

2.5 Classification Setup

As both datasets are multi-label, we employ one-vs-all settings for all models. A one-vs.-all solution consists of N (here, N = 6) separate binary classifiers, each answering a separate classification question during training [31].

We divide the datasets into stratified train (80%) and test (20%) splits based on emotions using random sampling [32]. For all 6 models, we will use the same train and test sets.

3 EXPERIMENTS

3.1 RQ1: How accurately can Pre-Trained Language Models classify emotions in software engineering texts compared to state-of-the-art model?

The results for emotion analysis on GitHub and Stack Overflow datasets, utilizing the PTMs are presented in Table 2. The table includes F1-scores for individual emotion classes, micro and macro-averaged F1-scores, and the performance improvement over the baseline SEntiMoji model. From the table, it is evident that all models demonstrate performance improvement for both datasets in most cases.

GitHub Dataset: DeBERTa attains the highest micro and macro-averaged F1-scores, exhibiting improvements of 15.07% and 16.79%, respectively. ALBERT performed worse than the models in both metrics. The two SE-specific models perform similar. Across all individual emotions, PTMs display improvement in 24 out of 36 instances. Surprise sees improvement for all six models, Love and Fear for five models, Anger and Sadness for three models, and Joy for two models. Breaking it down by individual models, BERT improves four emotions, RoBERTa five emotions, ALBERT three emotions, DeBERTa six emotions, CodeBERT three emotions, and GraphCodeBERT three emotions. Notably, DeBERTa demonstrates the most significant improvement in Anger (25.72%), RoBERTa in Love (20.60%), RoBERTa in Fear (48.66%), DeBERTa in Joy (8.69%), RoBERTa in Sadness (3.85%), and BERT in Surprise (39.49%).

Stack Overflow Dataset: RoBERTa achieves the highest micro-averaged F1-score, while DeBERTa achieves the highest macro-averaged F1-score, improving by 6.23% and 14.52%, respectively. Between the two SE-specific models, CodeBERT outperforms GraphCodeBERT in both micro and macro-averaged F1-score. Across all individual emotions, PTMs show improvement in 27 out of 36 instances. Love, Fear, and Joy improve for all six models, Anger for five models, Sadness for three models, and Surprise for one model. It's important to note that the Stack Overflow dataset is more imbalanced than the GitHub dataset, with Surprise distribution being only 1%, exhibiting the least improvement in all models. Breaking it down by individual models, BERT improves five emotions, RoBERTa five emotions, ALBERT four emotions, DeBERTa six emotions, CodeBERT four emotions, and GraphCodeBERT three

emotions. DeBERTa demonstrates the most significant improvement in Anger (3.60%), RoBERTa in Love (6.49%), DeBERTa in Fear (37.88%), ALBERT in Joy (47.0%), BERT and RoBERTa in Sadness (8.0%), and DeBERTa in Surprise (15.79%).

Error Analysis: To identify model mistakes, we conduct an error analysis. Due to space constraints, we focus exclusively on the GitHub benchmark which has gone prior error analysis [10]. To understand where the PTMs commonly make mistake, we examine 67 cases where all six models agreed, yet the ground truth differed (9 false positives, 58 false negatives), distributed across emotions. For assessment, we employ Novielli et al.'s [33] error categorization, using a thematic approach. An author initially mapped errors, and a senior undergraduate student reviewed and resolved disagreements through discussion.

Echoing Imran et al.'s observations [10], the prevalent categories are 'General Error' and 'Implicit Sentiment Polarity.' General errors, occurring 29/67 times, manifest when models misinterpret or struggle to comprehend lexical cues conveying emotions. For instance, the text "*Nice, this is more slick* 👍" is annotated as Joy. Another example, "*i'm actually surprised this didn't get flagged by the analyzer...*," annotated as Surprise, remains mispredicted by all models. The majority of Surprise (10/15), Joy (8/14), and Love (4/5) errors fall into the general errors category.

Implicit sentiment polarity errors occur 18/67 times. An example - "*Patiently waiting for any updates. [...]*" - is annotated as Sadness, with none of the models predicting it correctly. This category is particularly noticeable in Joy (6/14) and Sadness (5/11).

The third major error category is 'Figurative Language' (9/67), which occurs when users use humor, irony, sarcasm, metaphors, etc to convey emotion. This category is noticeable among negative emotions (Anger 5/13 and Fear 3/9). For example, the following utterance "*[...] I understand what you mean by "takeover" however it doesn't hurt to be a little more explicit.*" - annotated as Anger but none of the models were able to detect it correctly.

Of note, in 13/67 cases, the utterances contain emojis which contribute in expressing emotions. The models possibly fail to capture them. For instance, "*And yes, there should be tests* 😱 😱 😱" - annotated as Fear, none of the models predict it accurately.

RQ1 Takeaway: The transformer-based PTMs consistently outperform the state-of-the-art SEntiMoji model across all emotions for both datasets. DeBERTa and RoBERTa emerge as top performers. General PTMs outshine domain-specific ones in this context.

3.2 RQ2: Can integrating polarity features in the training improve Pre-Trained Language Models' ability for emotion classification?

In open-domain research, leveraging word polarity has proven effective for sentiment analysis [34–36]. We aim to explore its applicability to emotion classification within the SE domain.

3.2.1 Procedure. To integrate polarity features, we enhance PTMs through token-level attention, focusing on tokens associated with polarity words.

Initially, we employ natural language processing techniques, utilizing the Natural Language Toolkit (nltk) and SentiWordNet to extract word polarity. This involves a series of steps such as tokenization, part-of-speech tagging, and identification of words with discernible sentiment, resulting in a concise list of polarity words for each utterance.

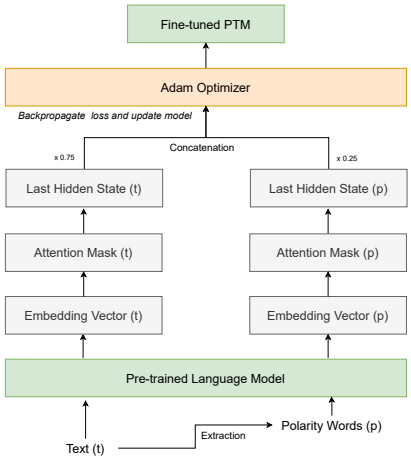

Figure 1: Fine-tuning procedure using token-level attention adjustment of polarity words.

Subsequently, the model architecture is fine-tuned at the attention level, with a focus on polarity words. Attention weights are adjusted to assign greater significance to tokens linked with polarity words. This involves a strategic blending of attention weights related to the primary input text and those corresponding to polarity words. The adjustment ensures a heightened emphasis on the embeddings of polarity words, achieved by multiplying attention weights for primary input text's hidden states by 0.75 and those for polarity words' hidden states by 0.25, followed by concatenation. This modification enhances the model's sensitivity to sentiment-carrying terms during training, allowing for improved discernment of subtle variations in sentiment expression. Refer to Figure 1 for a visual representation of the token-level attention adjustment procedure.

3.2.2 Results and Discussion. The results of emotion classification using polarity-enhanced PTMs are presented in Table 3, along with the percentage difference in performance against a state-of-the-art model and baseline PTMs. The findings show that polarity-enhanced PTMs outperform the state-of-the-art SEntiMoji model and baseline PTMs.

GitHub Dataset: SEntiMoji baseline: DeBERTa achieves the highest micro-averaged F1-score of 0.620, showing a 17.09% improvement. On the macro-averaged F1-score metric, BERT achieves the highest score of 0.621, reflecting a 19.28% improvement. Analyzing individual emotions, polarity-enhanced PTMs outperform SEntiMoji 33 out of 36 times, with improvements in Anger, Love, Fear, and Surprise for all six models, Joy for five models, and Sadness for four models. Anger achieves the highest F1-score of 0.588 by DeBERTa, marking a 27.88% improvement from SEntiMoji, Love F1-score 0.787 by RoBERTa (+22.63%), Fear F1-score 0.583 by BERT (+54.58%), Joy F1-score 0.633 by BERT (+13.85%), Sadness F1-score 0.674 by BERT (+7.07%), and Surprise F1-score 0.661 by BERT (+44.19%).

Table 3: Evaluation of Polarity-enhanced PTMs on the Emotions Dataset (F1-score).

Dataset	Model	Anger	Love	Fear	Joy	Sadness	Surprise	Micro Avg.	Macro Avg.
GitHub	BERT+Polarity	0.484	0.733	**0.583**	0.629	0.636	**0.661**	619	**0.621**
	SEntiMoji (+/-)	+5.19%	+14.13%	+54.58%	+13.11%	+1.00%	+44.19%	+16.98%	+19.28%
	BERT (+/-)	+13.71%	+0.26%	+6.94%	+5.42%	+4.41%	+3.37%	+5.99%	+5.04%
	RoBERTa+Polarity	0.475	**0.787**	0.538	0.583	0.654	0.598	0.603	0.606
	SEntiMoji (+/-)	+3.31%	+22.63%	+42.47%	+4.76%	+3.91%	+30.57%	+13.81%	+16.39%
	RoBERTa (+/-)	-8.12%	+1.68%	-4.16%	+11.89%	+0.06%	**+17.04%**	+4.94%	+2.75%
	ALBERT+Polarity	0.471	0.744	0.448	0.587	**0.674**	0.561	0.580	0.581
	SEntiMoji (+/-)	+2.30%	+15.92%	+18.66%	+5.46%	+7.07%	+22.34%	+9.49%	+11.54%
	ALBERT (+/-)	+5.45%	-1.16%	**+25.37%**	**+31.19%**	+6.83%	-6.82%	+7.86%	+7.65%
	DeBERTa+Polarity	**0.588**	0.680	0.507	**0.633**	0.654	0.623	**0.620**	0.614
	SEntiMoji (+/-)	+27.88%	+5.99%	+34.37%	+13.85%	+3.91%	+35.85%	+17.09	+18.01
	DeBERTa (+/-)	+1.72%	-7.51%	+6.48%	+4.74%	+1.92%	+1.89%	+1.75%	+1.04%
	CodeBERT+Polarity	0.565	0.691	0.576	0.530	0.607	0.640	0.595	0.601
	SEntiMoji (+/-)	+22.80%	+7.62%	+52.58%	-4.66%	-3.57%	+39.64%	+12.38%	+15.55%
	CodeBERT (+/-)	**+26.61%**	**+5.80%**	+5.08%	+2.36%	+2.68%	+11.54%	**+9.16%**	**+8.37%**
	GraphCodeBERT+Polarity	0.514	0.654	0.551	0.570	0.598	0.521	0.563	0.568
	SEntiMoji (+/-)	+11.63%	+1.91%	+45.94%	+2.42%	-5.01%	+13.70%	+6.22%	+9.09%
	GraphCodeBERT (+/-)	+7.84%	+3.58%	+8.70%	+3.19%	**+8.47%**	-9.86%	+2.52%	+3.38%
Stack Over-flow	BERT+Polarity	**0.785**	0.855	0.611	0.601	0.590	0.200	0.762	0.607
	SEntiMoji (+/-)	+3.44%	+4.36%	+42.59%	+38.05%	+6.15%	+10.0%	+6.68%	+14.55%
	BERT (+/-)	+2.02%	+0.40%	+12.04%	+0.73%	-1.71%	+20.0%	+1.0%	+3.17%
	RoBERTa+Polarity	0.777	**0.880**	0.650	0.594	0.575	**0.400**	**0.767**	**0.646**
	SEntiMoji (+/-)	+2.45%	+7.47%	+51.67%	+36.45%	+3.45%	+120.0%	+7.51%	+21.93%
	RoBERTa (+/-)	-1.11%	+0.92%	+11.94%	+0.50%	-4.21%	+140.0%	+1.20%	+7.78%
	ALBERT	0.777	0.844	**0.667**	0.598	**0.644**	0.167	0.757	0.616
	SEntiMoji (+/-)	+2.44%	+3.09%	+55.56%	+37.31%	+16.0%	-8.33%	+6.08%	+16.30%
	ALBERT (+/-)	+2.04%	-0.04%	+15.15%	-6.59%	+18.15%	+25.0%	+1.36%	**+10.23%**
	DeBERTa+Polarity	0.776	0.862	0.619	**0.643**	0.623	0.222	0.766	0.624
	SEntiMoji (+/-)	+2.29%	+5.27%	+44.44%	+47.74%	+12.21%	+22.22%	+7.36%	+17.84%
	DeBERTa (+/-)	-0.20%	+0.22%	+4.76%	+6.53%	+4.30%	+5.56%	+1.37%	+2.89%
	CodeBERT+Polarity	0.766	0.866	0.550	0.536	0.565	0.235	0.742	0.586
	SEntiMoji (+/-)	+0.95%	+5.70%	+28.33%	+23.0%	+1.65%	+29.41%	+3.99%	+10.64%
	CodeBERT (+/-)	-0.83%	+1.35%	-1.0%	+3.22%	+5.42%	+41.18%	**+1.91%**	+3.32%
	GraphCodeBERT+Polarity	0.727	0.848	0.524	0.583	0.565	0.167	0.732	0.569
	SEntiMoji (+/-)	-4.20%	+3.56%	+22.22%	+33.98%	+1.65%	-8.33%	+2.48%	+7.38%
	GraphCodeBERT (+/-)	-0.07%	+1.47%	+1.85%	+4.42%	+8.48%	+8.33%	+1.29%	+3.11%

PTM baseline: The improvement on PTM baseline ranges from 2.52% to 9.16% on micro-averaged F1-score and from 1.04% to 8.37% on macro-averaged F1-score. The most improved PTM is ALBERT in both micro and macro-averaged F1-score. enhancement is observed consistently across individual emotions, with Joy and Sadness improving for all six models, Anger and Fear for five models, Love for four models, and Surprise for four models. Noteworthy improvements include a 26.61% enhancement in Anger by CodeBERT, a 5.80% improvement in Love by CodeBERT, a 25.37% improvement in Fear by ALBERT, a 31.19% improvement in Joy by ALBERT, an 8.47% improvement in Sadness by GraphCodeBERT, and a 17.04% improvement in Surprise by RoBERTa.

Stack Overflow Dataset: SEntiMoji baseline: RoBERTa achieves the highest micro-averaged F1-score of 0.766, showing a 7.36% improvement; as well as and the highest-averaged F1-score of 0.646, reflecting a 21.93% improvement. Analyzing individual emotions, polarity-enhanced PTMs outperform SEntiMoji 33 out of 36 times, with improvements in Love, Fear, Joy, and Sadness for all six models, Anger for five models, and Surprise for four models. Anger achieved the highest F1-score of 0.777 by RoBERTa and ALBERT, marking a 2.44% improvement from SEntiMoji, Love F1-score 0.880 by RoBERTa (+7.47%), Fear F1-score 0.667 by ALBERT (+55.56%), Joy F1-score 0.643 by DeBERTa (+47.74%), Sadness F1-score 0.664 by BERT (+16.0%), and Surprise F1-score 0.400 by BERT (+120.0%).

PTM baseline: The improvement on PTM baseline ranges from 1.0% to 1.91% on micro-averaged F1-score and from 3.11% to 10.23% on macro-averaged F1-score. The most improved PTM is ALBERT in macro-averaged F1-score metric and CodeBERT in macro-averaged F1-score metric. While the improvement in Stack Overflow dataset against baseline PTMs is less compared to the GitHub dataset, this enhancement is observed consistently across individual emotions, with Surprise improving for all six models, Love, Fear and Joy for five models, Sadness for four models, and Anger for four models. Noteworthy improvements include a 2.04% enhancement in Anger by ALBERT, a 1.47% improvement in Love by GraphCodeBERT, a 15.15% improvement in Fear by ALBERT, a 6.53% improvement in Joy by DeBERTa, an 18.15% improvement in Sadness by ALBERT, and a 140.0% improvement in Surprise by RoBERTa.

The average PTM improvement in both dataset are similar to the general domain sentiment analysis results [34, 35].

Error Analysis: Similar to RQ1, we look into GitHub dataset for this case as well. In RQ1 error analysis, we observe that in 67 cases all models predict incorrectly. Out of those 67 cases, we find that 40 cases still produce erroneous results. However, in rest 27 cases, at least one model predict correctly. The most prominent resolved error categories are: 13/29 (44.82%) general errors, 9/18 (50%) implicit sentiment polarity, and 2/3 politeness. For example, "*i'm actually surprised this didn't get flagged by the analyzer...*," - this utterance is now correctly predicted by CodeBERT.

The least improved error category is 'Pragmatics'. Pragmatics is the error category when the classifiers deal with context information. That is often human annotators consider external facts for annotation. For example, consider the following utterance, *"[...] This change makes it the same as the line above and I don't see any reason to have two lines that are showing the same thing."* - is annotated as Anger as the commentator is annoyed/disagreed with the change in code. 6/7 (85.71%) Pragmatics related errors remains unresolved. Another least improved error category is 'Figurative Language' - 6/9 (66%) utterances predictions remained unchanged. For example, this following utterance, *"[...] We need to add this test coverage. It's just not 'urgent'.* 😐*"* - which expresses sarcasm and annotated as Anger. The models predict it incorrectly.

We observe that there still remains a considerable amount of utterances (10/13) that are misclassified contain emojis. For example, the previously mentioned utterance, "*And yes, there should be tests* 😱😱😱" - all models still predict incorrectly.

RQ2 Takeaway: Polarity-enhanced PTMs consistently outperform the SEntiMoji model and baseline PTMs in both datasets. Notable improvements are seen in micro- and macro-averaged F1-scores, along with substantial enhancements for individual metric across various models. While the findings indicate that integrating sentiment polarity improves PTM performance, it do not always help to capture context-dependent emotions.

4 DISCUSSION

In this section, based on our experiments, we summarize the key lessons learned and outline promising directions for future work related to emotion classification in SE.

4.1 Lessons Learned

Our study yields several key insights into the application of PTMs for emotion classification in SE texts. Firstly, general domain PTMs such as BERT, RoBERTa, and DeBERTa consistently outperform specialized models. This suggests that the pre-training objectives and textual domains carry greater relevance than task-specific tuning in the emotion classification task within SE text.

Secondly, the incorporation of polarity features during fine-tuning consistently enhances performance, emphasizing the value of sentiment awareness. However, challenges persist, especially with negative emotions like Anger and Fear, which pose difficulties in context understanding.

Thirdly, no single model excels across all emotions in all metrics, necessitating careful benchmarking and tradeoff analysis. Common error categories, such as implicit polarity, figurative language, and pragmatics, underscore the challenges in understanding contextual gaps in nuanced emotion recognition. Addressing these complexities requires further customization. For example, Imran et al. [37] addressed the issue of understanding figurative language in SE text using contrastive learning.

Lastly, persistent challenges arise in handling emojis, as models struggle to interpret emotive signals. Incorporating dedicated approaches to handle emojis may offer mitigation strategies.

4.2 Future Directions

Building on lessons learned, future work can focus on establishing more benchmark datasets, particularly emphasizing implicit and context-dependent emotions.

Investigating hierarchical emotion classification and joint modeling of related polarity dimensions merits investigation. Recognizing that emotion classification often treats each category separately, a hierarchical framework may enhance performance by first identifying broad emotional valence (positive, negative, or neutral) before classifying specific categories, as demonstrated by Li et al. [8].

In SE-specific models, exploring pre-training with polarity-word masking, emojis, and emoticons atop general word masking is warranted, given promising results in the general domain for sentiment analysis [38]. Research also suggests that aspect-based sentiment analysis (ABSA)-enhanced PTMs outperform baseline PTMs [35, 39]. As polarity-enhanced PTMs have already shown superiority over baseline PTMs, exploring ABSA-enhanced PTMs may further improve classifiers.

Additionally, investigating the multi-modal fusion of text and emojis cues during the pre-training/fine-tuning step could enhance the interpretation of emotive expressions. Exploring generative language models like GPT-4 and LLaMA for emotion detection using zero-shot and few-shot learning presents an intriguing direction. Recent advancements in these models enable synthesizing natural language responses, facilitating data augmentation and prompting techniques for improved classification [40–43]. Focusing on detecting the most relevant emotions that may harm productivity, such as Frustration and Disappointment, holds merit [44]. For example, a recent case study delved into understanding Frustration in OSS text using generative models in zero-shot learning [45].

5 RELATED WORK

On a large scale the related work can be divided into two parts: emotions in SE texts and PTMs in SE tasks.

5.1 Emotions in Text-Based Software Engineering Communication

Pletea et al. explored the emotions expressed by developers during security discussions on GitHub [46]. Ortu et al. analyzed the emotional expressions in JIRA issue comments and their impact on collaboration and software development outcomes [6]. Novielli et al. annotated a benchmark for emotion annotation in Stack Overflow using Shaver's [20] emotion model [7]. Calefato et al. developed a toolkit for emotion recognition from text [4]. Mäntylä et al. explored the possibilities of using Valence, Arousal, and Dominance (VAD) dimensions [47] for detecting burnout and productivity [48]. Islam et al. proposed MarValous – a machine learning method for

sensing emotions in the VAD space [49]. Werder et al. proposed a method for extracting emotions from GitHub repositories [50]. Destefanis et al. studied the measurement of affects in 370K GitHub issue comments to understand the emotional impact of commenters on the overall affect of GitHub issues [51]. Ortu et al. conducted a mining analysis on finding emotional state of contributors in GitHub repositories [52]. Venigalla et al. explored the emotions in software documentation [53]. Neupane et al. investigated emotion dynamics in software development [54].

Chen et al. utilized emojis for sentiment analysis and emotion detection in developers' communication through deep neural networks [5]. Rong et al. empirically studied emoji usage in software development communication, exploring frequency and diversity across different channels to understand developers' emotional expression [55]. Bleyl et al. applied a BERT-based emotion recognition model to Stack Overflow posts, aiming to identify developers' expressed emotions [11]. Imran et al. proposed an enhanced model based on Shaver's approach for software engineering text and introduced three data augmentation techniques to enhance Software Engineering emotion detection tool performance [10]. Tulli et al. [56] did a literature review on burnout in software engineering.

5.2 PTMs in Software Engineering Tasks

The utilization of PTMs has gained significant attention in software engineering text, offering a novel approach to address various challenges and enhance the performance of diverse tasks. One notable area of research involves the application of PTMs in code-related tasks. Researchers have explored the effectiveness of models like in understanding and generating code snippets. These studies aim to improve code completion, summarization, comprehension and documentation [14, 57–60]. Lanaguage models have shown promise in tasks related to software bugs and assessing code quality [13, 24, 61–64]. There also has been study on understanding code attentions in BERT [65]. Researchers also conduct studies on benchmarking PTMs on tag recommendation in Stack Overflow text, sentiment analysis in SE text and software aspect-based API review text [17, 66, 67].

In this study, we do comprehensive benchmark analysis of emotion classification task in SE text using PTMs and we also apply approaches in changing attention layer for further improving the performances.

6 THREATS TO VALIDY

Several limitations could affect the accurate interpretation of our results. We outline and detail each of these limitations below.

Construct validity. Construct validity assesses how well the study measures the concepts and constructs it aims to evaluate. While potential issues may arise from manually annotating the error analysis in RQ1, we mitigated this by employing thematic analysis to address any possible problems.

Internal validity. Internal validity focuses on the extent to which the study's findings can be attributed to the manipulation of the independent variable. Threats may arise from mistakes in our experiments, but we have provided a replication package and outputs for independent verifications. Another potential threat is the absence of cross-validation on smaller datasets, which we addressed by using stratified sampling for representativeness and an 80%-20% train-test split. There's also a potential threat related to data leakage in SEntiMoji models against the GitHub benchmark dataset, but this is unlikely as SEntiMoji was pre-trained before 2019 [5], and data points of the GitHub dataset came afterward [10]. However, verification of this is beyond the scope of our study given that not all pre-trained data is released by the authors [5].

External validity. External validity concerns the generalization of our study's findings to other settings and contexts. Our results may not extend beyond the specific models, datasets, and domains of GitHub and Stack Overflow. However, we utilized diverse pre-trained models and the most comprehensive datasets available in this field. Further investigation is necessary to validate our results beyond the datasets and pre-trained models used in our study.

7 CONCLUSION

This paper presented a comprehensive comparative analysis of state-of-the-art Pre-trained Transformer Language Models for emotion classification in software engineering texts. Through evaluation on two datasets, sourced from GitHub and Stack Overflow and annotated using Shaver's emotion model, we address two key research questions. First, we examined the accuracy of PTMs in classifying emotions compared to SEntiMoji, we found consistent enhancements with models like BERT, DeBERTa and RoBERTa achieving upto 16.79% improvements in terms of macro and micro-averaged F1 scores. The results showcase the prowess of general domain PTMs, with domain specific CodeBERT and GraphCodeBERT underperforming. Analysis of common errors revealed struggles in handling general comprehension, implicit polarity, figurative language and contextual pragmatics. Next, we incorporated polarity features during PTM finetuning and demonstrated additional gains over both SEntiMoji and baseline PTMs. This validates the benefit of augmenting with explicit polarity information for nuanced emotion classification. However, errors linked to figurative language and pragmatics persisted. In conclusion, this study provides a robust benchmark of diverse PTMs in a complex affective analysis task within software engineering, showcasing their effectiveness not only in sentiment analysis but also in fine-grained emotion recognition. These findings underscore the potential of PTMs in advancing empathetic software intelligence, with opportunities for further refinement to address contextual gaps.

REFERENCES

[1] N. Novielli and A. Serebrenik, "Emotion analysis in software ecosystems," in *Software Ecosystems: Tooling and Analytics*. Springer, 2023.
[2] B. Lin, N. Cassee, A. Serebrenik, G. Bavota, N. Novielli, and M. Lanza, "Opinion mining for software development: a systematic literature review," *ACM TOSEM*, vol. 31, 2022.
[3] A. Murgia, M. Ortu, P. Tourani, B. Adams, and S. Demeyer, "An exploratory qualitative and quantitative analysis of emotions in issue report comments of open source systems," *Empirical Software Engineering*, 2018.
[4] F. Calefato, F. Lanubile, N. Novielli, and L. Quaranta, "Emtk-the emotion mining toolkit," in *2019 IEEE/ACM 4th International Workshop on SEmotion*. IEEE, 2019.
[5] Z. Chen, Y. Cao, H. Yao, X. Lu, X. Peng, H. Mei, and X. Liu, "Emoji-powered sentiment and emotion detection from software developers' communication data," *ACM TOSEM*, 2021.
[6] M. Ortu, A. Murgia, G. Destefanis, P. Tourani, R. Tonelli, M. Marchesi, and B. Adams, "The emotional side of software developers in jira," in *Proceedings of the 13th MSR*, 2016.
[7] N. Novielli, F. Calefato, and F. Lanubile, "A gold standard for emotion annotation in stack overflow," in *2018 IEEE/ACM 15th MSR*. IEEE, 2018.

[8] J. Li, Y. Lei, S. Li, H. Zhou, Y. Yu, Z. Jia, Y. Ma, and T. Wang, "A two-stage framework for ambiguous classification in software engineering," in *2023 IEEE 34th ISSRE*. IEEE, 2023.

[9] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv*, 2018.

[10] M. M. Imran, Y. Jain, P. Chatterjee, and K. Damevski, "Data augmentation for improving emotion recognition in software engineering communication," in *Proceedings of the 37th ASE*, 2022.

[11] D. Bleyl and E. K. Buxton, "Emotion recognition on stackoverflow posts using bert," in *2022 IEEE International Conference on Big Data*. IEEE, 2022.

[12] 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," *arXiv*, 2019.

[13] A. Ciborowska and K. Damevski, "Fast changeset-based bug localization with bert," in *Proceedings of the 44th ICSE*, 2022.

[14] M. Ciniselli, N. Cooper, L. Pascarella, D. Poshyvanyk, M. D. Penta, and G. Bavota, "An empirical study on the usage of bert models for code completion," in *2021 IEEE/ACM 18th MSR*. IEEE, 2021.

[15] B. Lin, F. Zampetti, G. Bavota, M. Di Penta, M. Lanza, and R. Oliveto, "Sentiment analysis for software engineering: How far can we go?" in *Proceedings of the 40th ICSE*, 2018.

[16] O. B. Sghaier and H. Sahraoui, "A multi-step learning approach to assist code review," in *2023 IEEE SANER*. IEEE, 2023.

[17] T. Zhang, B. Xu, F. Thung, S. A. Haryono, D. Lo, and L. Jiang, "Sentiment analysis for software engineering: How far can pre-trained transformer models go?" in *2020 IEEE ICSME*. IEEE, 2020.

[18] J. Sarker, A. K. Turzo, and A. Bosu, "A benchmark study of the contemporary toxicity detectors on software engineering interactions," in *2020 27th APSEC*. IEEE, 2020.

[19] N. Cassee, "Sentiment in software engineering: detection and application," in *Proceedings of the 30th ACM Joint European Software Engineering Conference and Symposium on the Foundations of Software Engineering*, 2022.

[20] P. Shaver, J. Schwartz, D. Kirson, and C. O'connor, "Emotion knowledge: further exploration of a prototype approach." *Journal of personality and social psychology*, 1987.

[21] B. Felbo, A. Mislove, A. Søgaard, I. Rahwan, and S. Lehmann, "Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm," in *Proceedings of the 2017 Conference on EMNLP*, 2017.

[22] H. Batra, N. S. Punn, S. K. Sonbhadra, and S. Agarwal, "Bert-based sentiment analysis: A software engineering perspective," in *Database and Expert Systems Applications: 32nd International Conference, DEXA 2021*. Springer-Verlag, 2021.

[23] T. Wang, B. Sun, and Y. Tong, "Auto-absa: automatic detection of aspects in aspect-based sentiment analysis," *arXiv preprint arXiv:2202.00484*, 2022.

[24] X. Zhou, D. Han, and D. Lo, "Assessing generalizability of codebert," in *2021 IEEE ICSME*. IEEE, 2021.

[25] A. Karmakar and R. Robbes, "What do pre-trained code models know about code?" in *2021 36th IEEE/ACM International Conference on Automated Software Engineering (ASE)*. IEEE, 2021, pp. 1332–1336.

[26] P. He, X. Liu, J. Gao, and W. Chen, "Deberta: Decoding-enhanced bert with disentangled attention," *arXiv*, 2020.

[27] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," *arXiv*, 2019.

[28] Z. Feng, D. Guo, D. Tang, N. Duan, X. Feng, M. Gong, L. Shou, B. Qin, T. Liu, D. Jiang *et al.*, "Codebert: A pre-trained model for programming and natural languages," in *EMNLP 2020*, 2020.

[29] D. Guo, S. Ren, S. Lu, Z. Feng, D. Tang, S. Liu, L. Zhou, N. Duan, A. Svyatkovskiy, S. Fu *et al.*, "Graphcodebert: Pre-training code representations with data flow," *arXiv*, 2020.

[30] (2023) Hugging face. [Online]. Available: https://huggingface.co/

[31] (2023) One vs all. [Online]. Available: https://developers.google.com/machine-learning/crash-course/multi-class-neural-networks/one-vs-all

[32] Z. Botev and A. Ridder, "Variance reduction," *Wiley statsRef: Statistics reference online*, 2017.

[33] N. Novielli, D. Girardi, and F. Lanubile, "A benchmark study on sentiment analysis for software engineering research," in *Proceedings of the 15th MSR*, 2018.

[34] H. Tian, C. Gao, X. Xiao, H. Liu, B. He, H. Wu, H. Wang, and F. Wu, "Skep: Sentiment knowledge enhanced pre-training for sentiment analysis," *arXiv*, 2020.

[35] Y. Zhang, Y. Yang, B. Liang, S. Chen, B. Qin, and R. Xu, "An empirical study of sentiment-enhanced pre-training for aspect-based sentiment analysis," in *Findings of the ACL 2023*, 2023.

[36] P. Ke, H. Ji, S. Liu, X. Zhu, and M. Huang, "Sentilare: Sentiment-aware language representation learning with linguistic knowledge," in *Proceedings of the 2020 Conference on EMNLP*, 2020.

[37] M. M. Imran, P. Chatterjee, and K. Damevski, "Shedding light on software engineering-specific metaphors and idioms," *arXiv preprint arXiv:2312.10297*, 2023.

[38] J. Zhou, J. Tian, R. Wang, Y. Wu, W. Xiao, and L. He, "Sentix: A sentiment-aware pre-trained model for cross-domain sentiment analysis," in *Proceedings of the 28th international conference on computational linguistics*, 2020.

[39] H. Yang, C. Zhang, and K. Li, "Pyabsa: A modularized framework for reproducible aspect-based sentiment analysis," in *Proceedings of the 32nd ACM CIKM*, 2023.

[40] M. Bayer, M.-A. Kaufhold, and C. Reuter, "A survey on data augmentation for text classification," *ACM Computing Surveys*, 2022.

[41] J. Kocon, I. Cichecki, O. Kaszyca, M. Kochanek, D. Szydło, J. Baran, J. Bielaniewicz, M. Gruza, A. Janz, K. Kanclerz *et al.*, "Chatgpt: Jack of all trades, master of none," *Information Fusion*, 2023.

[42] B. Koptyra, A. Ngo, Ł. Radliński, and J. Kocoń, "Clarin-emo: Training emotion recognition models using human annotation and chatgpt," in *International Conference on Computational Science*. Springer, 2023.

[43] A. Nedilko, "Generative pretrained transformers for emotion detection in a code-switching setting," in *Proceedings of the 13th Workshop on Computational Approaches to Subjectivity, Sentiment, & Social Media Analysis*, 2023.

[44] M. R. Wrobel, "Emotions in the software development process," in *2013 6th International Conference on Human System Interactions (HSI)*. IEEE, 2013.

[45] M. M. Imran, P. Chatterjee, and K. Damevski, "Uncovering the causes of emotions in software developer communication using zero-shot llms," *arXiv preprint arXiv:2312.09731*, 2023.

[46] D. Pletea, B. Vasilescu, and A. Serebrenik, "Security and emotion: sentiment analysis of security discussions on github," in *Proceedings of the 11th working conference on mining software repositories*, 2014.

[47] J. A. Russell and A. Mehrabian, "Evidence for a three-factor theory of emotions," *Journal of research in Personality*, 1977.

[48] M. Mantyla, B. Adams, G. Destefanis, D. Graziotin, and M. Ortu, "Mining valence, arousal, and dominance: possibilities for detecting burnout and productivity?" in *Proceedings of the 13th MSR*, 2016.

[49] M. R. Islam, M. K. Ahmmed, and M. F. Zibran, "Marvalous: Machine learning based detection of emotions in the valence-arousal space in software engineering text," in *Proceedings of the 34th ACM/SIGAPP SAC*, 2019.

[50] K. Werder and S. Brinkkemper, "Meme: toward a method for emotions extraction from github," in *Proceedings of the 3rd International Workshop on SEmotion*, 2018.

[51] G. Destefanis, M. Ortu, D. Bowes, M. Marchesi, and R. Tonelli, "On measuring affects of github issues' commenters," in *Proceedings of the 3rd International Workshop on SEmotion*, 2018.

[52] M. Ortu, T. Hall, M. Marchesi, R. Tonelli, D. Bowes, and G. Destefanis, "Mining communication patterns in software development: A github analysis," in *Proceedings of the 14th PROMISE*, 2018.

[53] A. S. M. Venigalla and S. Chimalakonda, "Understanding emotions of developer community towards software documentation," in *2021 IEEE/ACM 43rd ICSE: Software Engineering in Society (ICSE-SEIS)*. IEEE, 2021.

[54] K. P. Neupane, K. Cheung, and Y. Wang, "Emod: An end-to-end approach for investigating emotion dynamics in software development," in *2019 IEEE ICSME*. IEEE, 2019.

[55] S. Rong, W. Wang, U. A. Mannan, E. S. de Almeida, S. Zhou, and I. Ahmed, "An empirical study of emoji use in software development communication," *Information and Software Technology*, vol. 148, 2022.

[56] T. R. Tulili, A. Capiluppi, and A. Rastogi, "Burnout in software engineering: A systematic mapping study," *Information and Software Technology*, vol. 155, 2023.

[57] S. Gao, C. Gao, Y. He, J. Zeng, L. Nie, X. Xia, and M. Lyu, "Code structure–guided transformer for source code summarization," *ACM Transactions on Software Engineering and Methodology*, vol. 32, no. 1, 2023.

[58] S. Haque, Z. Eberhart, A. Bansal, and C. McMillan, "Semantic similarity metrics for evaluating source code summarization," in *Proceedings of the 30th IEEE/ACM International Conference on Program Comprehension*, 2022.

[59] S. MacNeil, A. Tran, A. Hellas, J. Kim, S. Sarsa, P. Denny, S. Bernstein, and J. Leinonen, "Experiences from using code explanations generated by large language models in a web software development e-book," in *Proc. 54th ACM Tech. Symp. on CS Education V. 1*, 2023.

[60] S. Rai, R. C. Belwal, and A. Gupta, "A review on source code documentation," *ACM Transactions on Intelligent Systems and Technology*, vol. 13, 2022.

[61] P. Ardimento and C. Mele, "Using bert to predict bug-fixing time," in *2020 IEEE Conference on Evolving and Adaptive Intelligent Systems (EAIS)*, 2020.

[62] W.-Y. Wang, C.-H. Wu, and J. He, "Clebpi: Contrastive learning for bug priority inference," *IST*, vol. 164, 2023.

[63] A. Ali, Y. Xia, Q. Umer, and M. Osman, "Bert based severity prediction of bug reports for the maintenance of mobile applications," *JSS*, 2023.

[64] J. Keim, A. Kaplan, A. Koziolek, and M. Mirakhorli, "Does bert understand code?–an exploratory study on the detection of architectural tactics in code," in *European Conference on Software Architecture*. Springer, 2020.

[65] R. Sharma, F. Chen, F. Fard, and D. Lo, "An exploratory study on code attention in bert," in *Proceedings of the 30th IEEE/ACM ICPC*, 2022.

[66] C. Yang, B. Xu, J. Y. Khan, G. Uddin, D. Han, Z. Yang, and D. Lo, "Aspect-based api review classification: How far can pre-trained transformer model go?" in *2022 IEEE International Conference on SANER*. IEEE, 2022.

[67] J. He, B. Xu, Z. Yang, D. Han, C. Yang, and D. Lo, "Ptm4tag: sharpening tag recommendation of stack overflow posts with pre-trained models," in *Proceedings of the 30th IEEE/ACM ICPC*, 2022.