# A Socio-Technical Analysis of Developer Community Responses to Software Engineering Transformations in the Generative AI Era

**Syarif Romadloni[1*], Feddy Setio Pribadi[2]**

[1,2]Electrical Engineering Department, Faculty of Engineering, Universitas Negeri Semarang, Semarang, Central Java, Indonesia
E-mail: [1*]syarifroma@gmail.com, [2]feddy.setio@mail.unnes.ac.id

(Received: 30 Sep 2026, revised: 13 Oct 2026, accepted: 16 Oct 2026)

**Abstract**

The rapid integration of Generative AI (GenAI) into software engineering has reshaped developer practices, roles, and working relationships, making it harder to understand how developer perceptions evolve over time. This study analyzed 80,570 Reddit and X documents from November 2022 to July 2026 across three observation phases. The corpus included 71,778 English, 8,183 Indonesian, and 609 mixed-language documents, with 83.4% sourced from X. BERTopic identified discussion topics, RoBERTa-based models classified sentiment, and a three-model LLM-as-a-Judge panel supported semantic validation. The analysis yielded 79 canonical topics, revealing a shift from coding assistance toward model preferences and employment-related concerns. Positive sentiment remained most common, although its distribution differed across topics. Career & Employment Concerns increased from 21.23% to 43.87% of classified socio-technical discourse, reflecting growing attention rather than measured anxiety. Discussions of developer–AI practices increasingly emphasized delegation and agentic execution. The findings demonstrate the value of examining topic prevalence and sentiment separately within socio-technical software engineering discourse and may inform developer training and software engineering curricula.

**Keywords:** Generative Artificial Intelligence, Software Engineering, Developer Discourse, Socio-Technical Analysis, Sentiment Analysis.

## I. INTRODUCTION

Generative AI has evolved from a general-purpose assistive tool into a technology integrated into various work activities [1]. In software engineering, this shift is evident in the use of GenAI, which now supports activities ranging from code writing and testing to documentation [2]. The growing capabilities of Large Language Models (LLMs) have also expanded how developers interact with AI, from chat assistants and extended auto-completion to code explanation and support for more complex tasks [3]. As these developments have progressed, GenAI's position in the software developer workflow has also changed—from chat-based assistance for specific tasks to the emergence of agentic coding systems that support more autonomous execution of development tasks [4], [5].

Changes in GenAI's role in software development have also affected developer practices, including roles, skills, working relationships [6], and dynamics within open-source communities [7]. These developments highlight the need to examine not only the technical outcomes of GenAI use but also its integration into software development practices [3], [6]. From a socio-technical perspective, GenAI integration involves the interaction between technical aspects, such as development workflows and output quality, and social aspects, including collaboration, professional roles, skills, and employment-related concerns [6], [8]. Accordingly, this study examines developer discourse using five socio-technical dimensions encompassing workflow integration and adaptation, verification and output quality, career and employment concerns, role and skill adaptation, and emerging practices. These dimensions provide an operational framework for interpreting the issues discussed in online communities. A recent systematic literature review also identified inconsistencies in how AI-tool adoption in software development is defined and measured [9]. Examining online discussions across different periods offers a complementary perspective on changes in the issues and sentiment expressed toward GenAI.

Previous studies have examined GenAI through public discussions and software development practices. Research on online discourse has analyzed public attitudes toward

DOI: 10.34148/teknika.v15i3.1616
TEKNIKA, Volume 15(3), November 2026, pp. 529-540
ISSN 2549-8037, EISSN 2549-8045

CC BY SA

ChatGPT [10], and explored broader patterns of AI-related topics and sentiment [11], [12]. Studies focusing on software development have investigated programmers' perceptions of AI tools [13], examined changes in AI-assisted development through longitudinal IDE activity logs [5], and identified technical challenges discussed on Stack Overflow concerning OpenAI APIs [14]. Together, these studies provide valuable insights into public opinion, development practices, and technical discussions. However, their different analytical scopes leave room for a more integrated examination of how topics, topic-level sentiment, and socio-technical concerns in GenAI-related software development discussions change over time.

To address this gap, this study examines GenAI-related software development discussions on Reddit and X across three observation phases, covering November 2022 to December 2023 (Phase 1), January to December 2024 (Phase 2), and January 2025 to July 2026 (Phase 3). These phases provide a structured basis for temporal comparison rather than a fixed classification of GenAI development. The study addresses four research questions. RQ1. How have the topics in GenAI-related software development discussions changed across the observation period? RQ2. How has sentiment within these topics changed across the observation period? RQ3. How have socio-technical concerns in these discussions changed across the observation period? RQ4. How have developer–AI practice orientations evolved in online discussions over the observation period? This study makes three contributions. First, it examines changes in GenAI-related software development discourse across three observation phases using Reddit and X data. Second, it integrates topic modeling, sentiment analysis, and LLM-based semantic validation to examine changes in discussion patterns. Third, it applies a socio-technical framework to interpret the issues and developer–AI interaction patterns discussed across the three phases.

## II. RESEARCH METHOD

This study employs computational text analysis supported by socio-technical interpretation to examine changes in GenAI-related software development discussions over time. Figure 1 presents the overall research workflow.

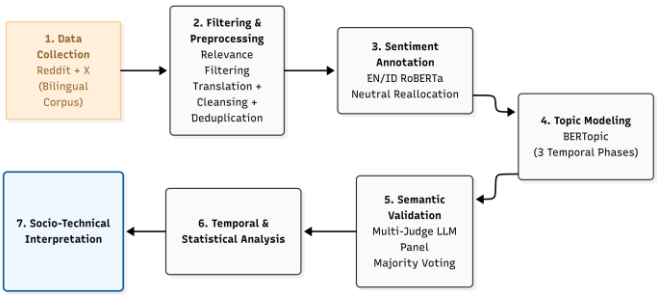



Figure 1. Research Method Flow

Figure 1 shows the research workflow, beginning with data collection, preprocessing, sentiment analysis, topic modeling, and semantic validation, followed by temporal analysis and socio-technical interpretation. The following subsections describe these stages.

### A. Data Collection and Preprocessing

Adopting the design framework of a previous study [15], data were collected from Reddit and X using English and Indonesian keywords related to GenAI and software development. The collection covered three observation phases: Phase 1 (November 2022 to December 2023), Phase 2 (January to December 2024), and Phase 3 (January 2025 to July 2026). Table 1 presents representative search keywords and the complete list of targeted subreddits across the observation period. X data were retrieved using the twscrape library through monthly search windows, while Reddit posts and comments were collected using the PullPush.io API for Phases 1 and 2 and the Arctic Shift API for Phase 3.

Table 1. Representative Search Keywords and Targeted Subreddits

| Search Component | Keywords or Communities |
|---|---|
| X -Keywords | ChatGPT write code; ChatGPT ngoding; ChatGPT hallucination code; AI gantiin programmer; github copilot; pakai copilot; cursor ai; developer workflow; ngoding pakai AI; copilot productivity; coding agent; agen AI ngoding; vibe coding; autonomous coding; developer career anxiety; loker IT sepi |
| Reddit-Keywords | programming with chatgpt; copilot worth it; agentic coding; AI replacing programmers |
| Sub-Reddit | r/artificial; r/ChatGPT; r/ClaudeAI; r/coding; r/compsci; r/cscareerquestions; r/developersIndia; r/ExperiencedDevs; r/githubcopilot; r/javascript, r/learnprogramming; r/LocalLLaMA; r/MachineLearning; r/OpenAI; r/programming; r/Python; r/singularity; r/softwareengineering; r/vscode; r/webdev |

During data collection, we excluded retweets from X searches, while retaining replies and Reddit comments to capture community discussions. We addressed potentially irrelevant, automated, and promotional content through subsequent document- and topic-level filtering rather than systematic bot-account detection. Consequently, some promotional or automated content may remain in the corpus.

After initial merging and deduplication, 186,037 documents were obtained for relevance screening. To reduce out-of-domain content, relevance filtering was performed using a zero-shot BART-MNLI classifier and an iterative topic-filtering approach [16]. To support bilingual classification, Indonesian and mixed-language documents were translated into English using MarianMT as a pivot language [17]. We then excluded topic clusters associated with irrelevant discussions, spam, and generic promotional content after inspection. This initial filtering stage removed 26,031

documents, leaving 160,006 documents for further preprocessing.

Subsequent preprocessing involved additional duplicate removal, text cleaning, protection of domain-specific entities, vocabulary normalization, and language identification using langid [18], Document-level filtering was also applied to remove unsuitable texts, producing a cleaned corpus of 80,570 documents for subsequent analysis. Table 2 presents sample documents together with platform metadata and detected language.

Table 2. Examples of Raw Documents

| ID | Platform | Text | Lang |
|---|---|---|---|
| 18140 | x | Untuk nulis coding tentu sangat enak. Tinggal ... | id |
| 8503 | x | Any chance we can jump on a call about this so... | en |
| 20257 | reddit | AI is not at the level where it can do this we... | en |
| 10359 | reddit | Well, ChatGPT just regurgitates google searche... | en |

The sample documents in Table 2 illustrate the informal and multilingual characteristics of the collected text. Figure 2 shows how raw text was transformed into a semantically clean representation while preserving technical terms and contextual information relevant to subsequent analysis.

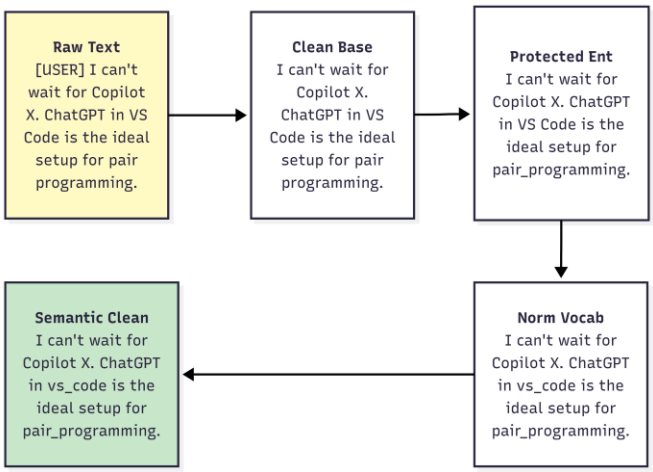



Figure 2. Text Preprocessing Example

These preprocessing procedures produced a final corpus of 80,570 documents for subsequent analysis. Table 3 summarizes the cleaned corpus distribution by platform, language, and observation phase.

Table 3. Corpus Distribution by Platform, Language, and Temporal Phase.

| Phase | Platform | English | Indonesian | Mixed |
|---|---|---|---|---|
| 1 | Reddit | 3593 | 1 | 3 |
| | X | 17372 | 359 | 24 |
| 2 | Reddit | 1755 | 0 | 0 |
| | X | 18251 | 5249 | 376 |
| 3 | Reddit | 8057 | 0 | 0 |
| | X | 22750 | 2574 | 206 |

## B. Sentiment Analysis

This stage implemented a model to label sentiment in the bilingual corpus. Sentiment inference was run in parallel using two fine-tuned models based on the RoBERTa architecture [19]. Operationally, through the Hugging Face computing ecosystem, English-language documents were analyzed using the cardiffnlp/twitter-roberta-base-sentiment-latest model, optimized for social media discourse [20], while Indonesian-language documents were classified using the w11wo/indonesian-roberta-base-sentiment-classifier model. Both models were implemented using the Hugging Face Transformers library through AutoTokenizer and AutoModelForSequenceClassification. We classified English and mixed-language documents using semantic-clean text, whereas we classified Indonesian documents using preprocessed text with protected technical entities. We assigned mixed-language documents (n = 609) to the English model. Inference was performed without additional fine-tuning, using a batch size of 64 and a maximum sequence length of 128 tokens.

To obtain a probability value (confidence score) for each sentiment class (positive, neutral, negative), the models' raw outputs (logits) were converted using the softmax function. The probability $P(y = i|x)$ of document x belonging to class i is formulated in (1).

$$\mathrm{P(y = i|x)} = \frac{\mathrm{e^{z_i}}}{\sum_{\mathrm{j=1}}^{\mathrm{K}} \mathrm{e^{z_j}}} \quad (1)$$

Where $z_i$ is the logit value for class $i$, and $K = 3$ represents the total number of sentiment classes. Computationally, this transformation was implemented in Python using scipy.special.softmax. The class with the highest probability was assigned as the initial sentiment prediction.

Furthermore, given the challenges of identifying neutral sentiment in BERT-based classification models [21], as well as findings that pretrained transformers can produce overconfident predictions [22]. We propose a sensitivity analysis for the two-stage neutral reallocation procedure. A document initially classified as neutral will be reallocated only if it meets two conditions: its probability falls below the neutral confidence threshold, and the higher of its positive or negative probabilities meets the minimum threshold for reallocation. Afterward, the document will be labeled as positive or negative, as appropriate. If it does not meet these conditions, the document remains neutral. Five pairs of thresholds, listed in Table 4, will be compared to the original argmax baseline against human-validated reference labels. Macro-F1 will be used as the primary metric, supported by accuracy, class-wise F1, and reallocation statistics. The Results and Discussion section will discuss the findings and their implications for threshold selection.

Table 4. Threshold Settings for Sensitivity Analysis

| Setting | Neutral threshold | Min. probability |
|---|---|---|
| 1 | 0.50 | 0.20 |
| 2 | 0.55 | 0.23 |
| 3 | 0.60 | 0.25 |
| 4 | 0.65 | 0.28 |
| 5 | 0.70 | 0.30 |

### C. Topic Modeling

Discourse themes were extracted by fitting a separate BERTopic model to each of the three temporal phases [23] This approach was selected to identify themes within each phase before examining their similarities and differences across phases. Topic clusters were therefore estimated from the documents within each period, with cross-phase correspondences established after topic extraction. BERTopic was selected because its topic representations are based on contextual embeddings, which have been shown in other technical discourse domains to produce more coherent and interpretable topics than word-frequency-based approaches such as LDA [24], [25]. Semantic representations were constructed using the Hugging Face embedding model BAAI/bge-m3. The embedding dimensions were reduced using UMAP [26], and clustered with HDBSCAN [27] to identify topic groups and outlier documents. Topic keywords were generated using c-TF-IDF with frequent-word reduction enabled. Outlier documents were subsequently reassigned based on c-TF-IDF similarity and embedding proximity, while highly similar topics were merged through agglomerative topic reduction. The quality of the resulting topics was assessed using NPMI, $C_v$, and topic diversity [28].

Preliminary inspection identified mixed English and Indonesian keywords that were difficult to interpret, particularly in Phase 2. Phase 2 was therefore rerun using cleaned, English-translated text as a targeted adjustment to improve readability and lexical consistency in its topic representations, while Phases 1 and 3 retained their original-language inputs. Topics were subsequently aligned across phases using keyword overlap and semantic similarity. Related topics were mapped to shared identifiers, which were then used to compare topic prevalence and examine temporal patterns across the three phases.

### D. LLM-as-a-Judge-Based Semantic Validation

Semantic validation of topic representations was performed using an LLM-as-a-Judge approach to assess label fit, document content, domain relevance, and topic coherence [29], [30]. Repeated assessment using a shared rubric was intended to reduce the manual evaluation workload [31], [32]. A DeepSeek generator received c-TF-IDF keywords and up to three representative documents per topic, each truncated to 400 characters, and was prompted to return a topic label of up to five words and an Indonesian socio-technical synthesis of two to four sentences in JSON format. A panel comprising DeepSeek, Qwen, and Kimi evaluated these outputs to limit dependence on a single evaluator [33]. Each judge received the same prompt, scored the four criteria from 1 to 5, and returned an accept, revise, or reject verdict with a brief rationale. The prompt instructed judges to disregard synthesis length when scoring. Generator and judge temperatures were 0.3 and 0.1, respectively, with a maximum output length of 2,048 tokens. The protocol specified three runs per judge, with document order shuffled between runs. Figure 3 summarizes the validation workflow.

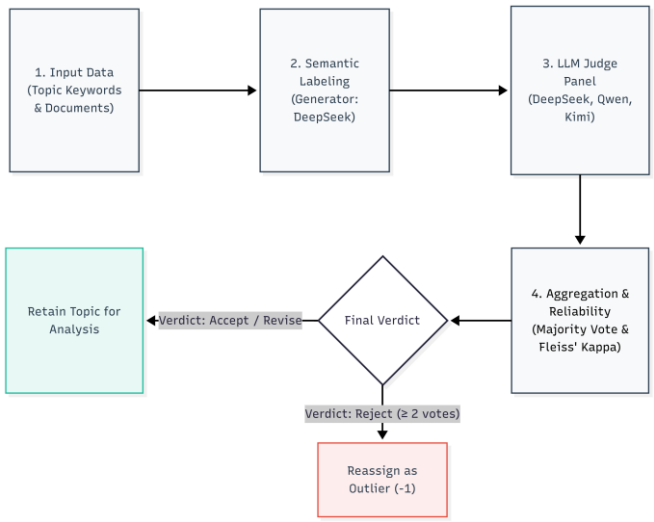



Figure 3. LLM-as-a-Judge Semantic Validation Pipeline

Within this workflow, a run returned reject when domain relevance or coherence scored 2 or lower; otherwise, it returned revise when label fit or synthesis grounding scored 2 or lower and accept in the remaining cases. Each model contributed its modal verdict across valid runs, with tied modes resolved by selecting the first occurring mode. A topic was rejected when at least two models voted reject; otherwise, any revise vote flagged it for revision, and the remaining topics were accepted. Accepted and revision-flagged topics were retained for temporal analysis, while documents belonging to rejected topics were reassigned to the outlier category (−1). Inter-model agreement was assessed separately using Fleiss' Kappa across the archived model-level verdicts for all evaluated topics, including rejected topics. The three models were treated as raters, and accept, revise, and reject as nominal categories [34].

### E. Temporal Evolution and Socio-Technical Interpretation

Temporal evolution was analyzed by aligning validated topics across phases using a hybrid similarity measure combining Jaccard similarity of representative c-TF-IDF terms (weight = 0.30) and cosine similarity of BGE-M3 semantic embeddings (weight = 0.70). A similarity threshold of 0.45 was applied to form canonical topic clusters. The prevalence of each canonical topic was compared across the three observation phases to identify patterns of emergence, persistence, growth, and decline, using a minimum prevalence threshold of 0.005 and at least 30 documents. Changes in monthly topic prevalence were examined using Binary Segmentation and Dynamic Programming with an L2 cost function and two predefined changepoints [35], A sensitivity


CC BY SA

analysis was conducted using penalty-based Binary Segmentation and PELT, with penalty factors ranging from 0.01 to 0.20, to assess the stability of the detected changepoints. Detected changepoints were interpreted as shifts in discussion patterns rather than causal effects of specific events. Topic-level sentiment changes were analyzed using Net Sentiment, calculated as the difference between the proportions of positive and negative documents for each canonical topic within each phase, as formulated in (2):

$$NS_{t,p} = P_{pos,t,p} - P_{neg,t,p} \quad (2)$$

where $\mathrm{NS}_{t,p}$ denotes the Net Sentiment of canonical topic $t$ in phase $p$, while $P_{pos,t,p}$ and $P_{neg,t,p}$ represent the proportions of positive and negative documents, respectively, relative to all documents assigned to that topic within the phase. Positive values indicate a higher proportion of positive than negative discourse, whereas negative values indicate the opposite.

For the socio-technical interpretation, the coding framework was informed by prior studies on human and technical changes associated with GenAI-enabled software engineering [6], [36], [37]. Each canonical topic was subsequently classified according to the socio-technical dimension that most closely reflected its primary discourse focus. The five dimensions and their corresponding operational definitions are summarized in Table 5.

Table 5. Operational Definitions of Socio-Technical Dimensions

| Category | Operational Definition |
|---|---|
| Workflow Integration & Adaptation | Discussions of how GenAI is integrated into or changes software-development workflows, task organization, tool use, and automation. |
| Verification & Output Quality | Discussions concerning verification, correctness, reliability, debugging, review, or the quality of AI-generated outputs. |
| Career & Employment Concerns | Discussions concerning job security, employability, hiring, displacement, career prospects, or AI-related effects on developer employment. |
| Role & Skill Adaptation | Discussions of changes in developer responsibilities, competencies, learning requirements, or professional skills associated with GenAI use. |
| Emerging Practices | Discussions of newly emerging GenAI-enabled development practices or interaction patterns beyond conventional AI-assisted coding. |

As summarized in Table 5, the socio-technical dimensions were used to capture the primary issues reflected in developer discourse rather than to infer direct behavioral or psychological changes among developers. Practice orientation was then used to characterize the form of developer–AI interaction represented by each canonical topic. This framework was adapted from the human–AI collaboration continuum described in [38], while the Agentic-oriented category was further informed by research on autonomous LLM-based software engineering agents [39]. The operational definitions of the practice-orientation categories are presented in Table 6.

Table 6. Operational Definitions of Practice-Orientation Categories

| Category | Operational Definition |
|---|---|
| Assistant-oriented | AI supports a specific and bounded task under direct human instruction, while the developer retains primary control. |
| Collaborative | Human and AI contribute iteratively to the same development activity and share responsibility for task completion. |
| Delegative | The developer delegates a broader task to AI while retaining responsibility for defining goals, supervision, or reviewing the output. |
| Agentic-oriented | AI performs multi-step development activities with greater operational autonomy, including planning, tool use, or task execution with limited human intervention. |

Practice-orientation coding was restricted to topics that explicitly reflected developer–AI interaction patterns, while topics with insufficient practice-related evidence were excluded from this analysis. Each canonical topic was assessed using its validated topic label, representative c-TF-IDF terms, and supporting documents. To evaluate coding reliability, two coders independently classified a subset of canonical topics selected across different prevalence levels. Inter-rater agreement was assessed separately for socio-technical dimensions and practice orientations using Cohen's kappa. Disagreements were reviewed by the primary researcher, while the remaining topics were classified using the same operational framework. The final mappings were applied to the complete set of canonical topics for subsequent temporal and socio-technical analyses.

## III. RESULT AND DISCUSSION

### A. Temporal Evolution of Developer Discourse

To trace how developer discourse concerning Generative AI evolved throughout the study period, the analysis focused on changes in relative prevalence and topic persistence over time. This approach ensured that proportional measures accounted for differences in document volume across periods.

Based on discourse-theme extraction, the BERTopic identified 89 topics in Phase 1, 99 in Phase 2, and 109 in Phase 3. Initial HDBSCAN clustering produced outlier rates of 67.44%, 69.74%, and 72.88%, respectively. Following outlier reassignment using c-TF-IDF similarity and semantic embeddings, these rates decreased to 1.87%, 2.43%, and 2.22%. This substantial reduction reflects the reassignment of initially unclustered documents to semantically similar topics rather than improvements in the original clustering performance. Topic quality was therefore evaluated

separately using coherence, topic diversity, and LLM-based semantic validation. The results are summarized in Table 7.

Table 7. Topic Modeling Quality and LLM-as-a-Judge Semantic Validation

| Phase | Initial Topic (Outlier) | $C_v$ | Diversity | LLM Verdict (A / Rv / Rj) |
|---|---|---|---|---|
| 1 | 89 (1.87%) | 0.4646 | 0.7461 | 23 / 51 / 15 |
| 2 | 99 (2.43%) | 0.4681 | 0.7768 | 19 / 49 / 31 |
| 3 | 109 (2.22%) | 0.4747 | 0.7615 | 36 / 54 / 19 |

The metric measurements in Table 7 show that the resulting topic representations had topic-diversity values of 0.7461–0.7768, while $C_v$ coherence scores ranged from 0.4646 to 0.4747, indicating relatively consistent coherence across phases. We then assessed the topics using an LLM-as-a-Judge panel. Only 19-36 topics per phase received outright Accept verdicts, while the high number of Revise and Reject verdicts shows that many topics still had semantic quality issues and required further evaluation. Inter-judge agreement across 297 evaluated topics was substantial (Fleiss' $\kappa$ = 0.669), with unanimous verdicts on 202 topics (68.0%). Table 8 presents representative topics and their final panel verdicts.

Table 8. Representative Topic Extraction and Panel Verdicts.

| Topic Label | Representative Words | Representative Document | Panel Verdict |
|---|---|---|---|
| ChatGPT for Coding and Debugging | write, coding, use chatgpt, debugging, code, help | I am just learning but one meta-solution for using ChatGPT for helping with any task is to first break the task down into specific roles... | Accept |
| AI Tools for Research & Teaching | learning, students, academia, tools, copilot, research | Don't use auto completion AI tools. github_copilot/tabnine is not healthy for you if you are learning. It is of course easy to find solutions... | Revise |
| Debugging Legacy Code Under Pressure | legacy, pressure, fix, bugs, privacy, gpt4 | When they have code_completion for their Jetbrains plugin, I'll check them out. Fwiw, their Jetbrains plugin says they use GPT3 4, so I'm wondering... | Reject |

Through this validation process, topics receiving a Reject verdict were reassigned to the outlier category, leaving 74 valid topics in Phase 1, 68 in Phase 2, and 90 in Phase 3 as the basis for further analysis. Because the topics were extracted separately in each phase, alignment was necessary to compare the same themes consistently. Therefore, all validated topics were aligned using a hybrid similarity measure combining Jaccard similarity of c-TF-IDF terms and cosine similarity of BGE-M3 embeddings, producing 79 canonical topics as the primary units of analysis. Based on their prevalence patterns, these topics were classified into seven lifecycle categories: negligible, disappearing, emerging, episodic, persistent, declining, and growing. This classification enables tracing themes that appeared, persisted, declined, or grew throughout the study period, as shown in Table 9.

Table 9. Distribution and Lifecycle of Canonical Topics Across Three Temporal Phases.

| Lifecycle Category | Topics | Canonical Representation Example | P1% | P2% | P3% |
|---|---|---|---|---|---|
| Negligible | 33 | (Marginal signals / fragmented noise) | < 0.5 | < 0.5 | < 0.5 |
| Disappearing | 21 | AI Code Completion Tools | 2.30 | 8.18 | 0.00 |
| Emerging | 7 | AI Model Preferences for Development | 0.49 | 6.67 | 13.40 |
| Episodic | 4 | Lifelong Learning in Coding | 2.06 | 0.00 | 8.97 |
| Persistent | 2 | GitHub Copilot Autocomplete Experience | 1.48 | 4.02 | 1.05 |
| Growing | 6 | AI Replacing Developer Jobs | 8.02 | 20.58 | 28.47 |
| Declining | 6 | ChatGPT for Coding and Debugging | 13.97 | 3.13 | 1.11 |

Table 9 shows that 33 of the 79 canonical topics fell into the negligible category, indicating that these themes had relatively low prevalence across the observation phases. Outside this category, various patterns of discourse change emerged across phases. ChatGPT for Coding and Debugging showed a declining pattern, with prevalence falling from 13.97% in Phase 1 to 1.11% in Phase 3. Conversely, AI Replacing Developer Jobs showed a growing pattern, with prevalence increasing from 8.02% to 28.47%. An emerging pattern was also observed for AI Model Preferences for Development, which increased from 0.49% in Phase 1 to 13.40% in Phase 3. Meanwhile, GitHub Copilot Autocomplete Experience appeared in all three phases and was categorized as persistent, indicating this topic's continuity in developer discourse. Lifelong Learning in Coding showed an episodic pattern, reappearing in Phase 3 after not being identified in Phase 2. AI Code Completion Tools showed a disappearing pattern, with prevalence reaching 0.00% in Phase 3.

The topic-change patterns show that developer discourse underwent shifts in thematic composition across phases and

did not develop linearly. In the early phase, discussions focused more on using AI to support coding activities and solve technical problems. Over time, these discussions declined, while attention to model preferences and AI's implications for developer jobs increased. This shift is reflected in discourse excerpts found in each phase. In the early phase, for example, the ChatGPT for Coding and Debugging topic contained extensive discussion of strategies for using AI to complete technical tasks, as shown in the following quotation: "I am just learning but one meta-solution for using ChatGPT for helping with any task is to first break the task down into specific roles or functions. software_development might be template, code, evolve, and test…." By contrast, in the final phase, discourse increasingly highlighted the possibility of replacing developer roles, as reflected in a discussion suggesting that AI agents could shift software development jobs toward supervising and coordinating AI systems while potentially reducing the need for other developer roles. These two examples show that changes in topic prevalence were also reflected in the content of community conversations. Discourse that initially focused on using AI to support coding activities developed into broader discussions of model selection and the potential influence of Generative AI on developers' roles and jobs.

In addition to changes in topic composition, this study examined shifts in monthly topic prevalence using changepoint detection. Under the fixed two-breakpoint configuration, Binary Segmentation and Dynamic Programming identified changes around February and December 2024. A sensitivity analysis using Binary Segmentation and PELT with different penalty settings identified two changepoints in January 2024 and January 2025 for penalty factors ranging from 0.02 to 0.20. At the lowest penalty (0.01), both methods detected six changepoints, including the two main transitions. These findings suggest that the main changes remained relatively stable near the research-phase boundaries, although the number of detected changepoints depended on the penalty setting. The identified changes indicate shifts in discussion patterns rather than causal effects of specific events. Figure 4 illustrates the changepoint detection results.

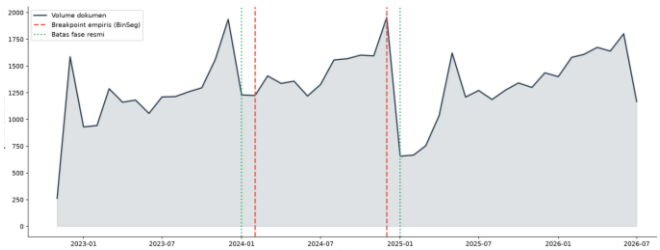



Figure 4. Changepoint Timeline

Overall, developer discourse concerning Generative AI shifted focus across the three research phases. Discussions initially focused on using AI to support coding activities and later expanded to broader issues, particularly model development and its implications for developers' roles and jobs.

**B. Semantic and Sentiment Shifts Across GenAI Phases**

Changes in developer discourse were examined not only through topic composition but also through sentiment dynamics within the identified topics. Before analyzing these changes, the confidence-based neutral reallocation procedure was evaluated against the original argmax predictions using a human-labeled sample focused on initially neutral documents. Five threshold configurations were compared using macro-F1, accuracy, and reallocation statistics. Table 10 presents the results.

Table 10. Neutral Reallocation Sensitivity

| Threshold (Neutral/Polarity) | Reallocated n (%) | Accuracy | Macro-F1 |
|---|---|---|---|
| Argmax baseline | 0 (0.0%) | 0.560 | 0.239 |
| 0.50 / 0.20 | 0 (0.0%) | 0.560 | 0.239 |
| 0.55 / 0.23 | 14 (16.7%) | 0.583 | 0.381 |
| 0.60 / 0.25 | 0 (0.0%) | 0.560 | 0.239 |
| 0.65 / 0.28 | 8 (9.5%) | 0.536 | 0.311 |
| 0.70 / 0.30 | 10 (11.9%) | 0.464 | 0.235 |

Table 10 shows that, among 84 human-labeled documents initially classified as neutral, the 0.55 neutral confidence threshold and 0.23 minimum polarity probability achieved the highest macro-F1 score (0.381) among the evaluated configurations, compared with 0.239 for the argmax baseline. Accuracy also increased from 55.95% to 58.33%, with 14 documents (16.7%) reallocated, of which five matched the human reference labels. These results support the selected configuration within the evaluated sample, although its effectiveness cannot be generalized to the entire corpus. Table 11 illustrates the application of the selected thresholds to representative documents drawn from the full corpus.

Table 11. Neutral Class Reallocation

| Doc | Confidence (Neutral) | Positive Prob. | Negative Prob. | Final Class |
|---|---|---|---|---|
| Agreed, only through verifiable result we can unlock parallel and near instant code generation. Reviewing code is a... | 0.55 | 0.33 | 0.11 | Positive |
| A user expressed concern that AI could replace programmers who mainly turn requirements into code, although software engineers may still be needed. | 0.55 | 0.09 | 0.36 | Negative |

DOI: 10.34148/teknika.v15i3.1616
TEKNIKA, Volume 15(3), November 2026, pp. 529-540
ISSN 2549-8037, EISSN 2549-8045

Table 11 illustrates how documents initially classified as neutral were reassigned to positive or negative sentiment based on the selected confidence thresholds. To examine the effect of this adjustment on temporal sentiment patterns, Figure 5 compares sentiment distributions before and after neutral reallocation across the three observation phases.

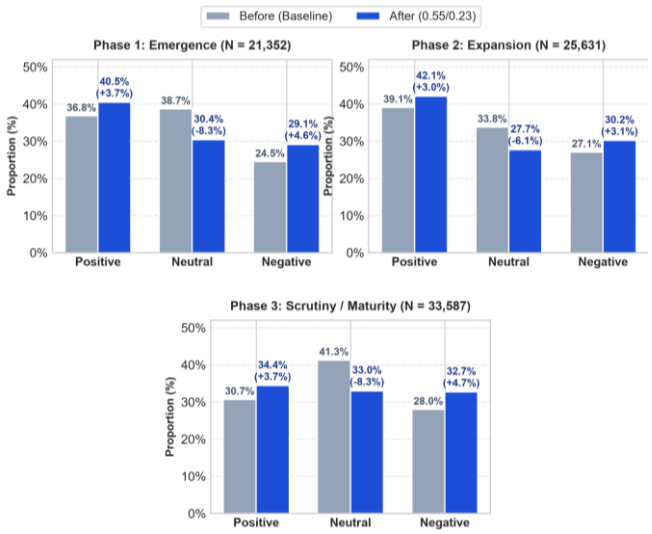



Figure 5. Sentiment Distribution After Confidence-Based Neutral Reallocation.

Figure 5 shows that neutral sentiment decreased after reallocation from 38.7% to 30.4% in Phase 1, 33.8% to 27.7% in Phase 2, and 41.3% to 33.0% in Phase 3, while positive and negative proportions increased. Positive sentiment remained the most prevalent category, accounting for 40.5%, 42.1%, and 34.4% across the three phases, respectively, following the same temporal pattern as the baseline. Nevertheless, these shifts show that developers' perceptions of Generative AI did not remain entirely constant throughout the study period.

For a clearer picture, the documents below represent the three sentiment categories. Positive-sentiment documents generally highlighted the benefits of Generative AI in supporting developers' work, for example: "Hey fellow programmers! I've been experimenting with AI tools to enhance my coding workflow, and it's been a game-changer. From autocomplete suggestions to generating boilerplate code, AI is helping me write cleaner, more efficient code faster than ever. Has anyone else tried integrating AI into...". In contrast, negative-sentiment documents contained more criticism or concern about AI use, such as: "2 years ago, I used to say my job is completely safe from AI. 'How could AI code better than me?' Oh, was I so naive. Every week I meet devs (especially Senior ones) who hesitate to try github_copilot, Claude, or Cursor. Yes it might not...". Neutral documents, meanwhile, tended to contain questions, information, or usage experiences without strong positive or negative sentiment, as shown in the quotation: "What industry are you working in? If your team is capable of developing shippable features with ChatGPT, then it seems the software being developed is not that complex. For the industry that I work in, even if I were to use ChatGPT/copilot to develop features, I would still need to do a lot of...". These examples show that the three sentiment categories are directly reflected in the content of community conversations and not merely in the aggregate classification results.

These document examples show that the sentiment differences identified in the analysis were visible not only in the data distribution but also in the content of developer conversations. To determine whether sentiment patterns differed across topics, the subsequent analysis compared Net Sentiment across the three phases, calculated as the difference between the proportions of positive and negative documents within each canonical topic. Table 12 presents the Net Sentiment trajectories of five representative canonical topics.

Table 12. Net Sentiment Trajectories Across Representative Canonical Topics.

| Canonical Topic | P1 | P2 | P3 | Δ P3 -P1 |
|---|---|---|---|---|
| AI Replacing Developer Jobs | -13.96 | 18.43 | 24.47 | 38.43 |
| ChatGPT for Coding and Debugging | 2.67 | 50.37 | -9.16 | -11.83 |
| Copilot Chat Quality Gap | 22.99 | -2.76 | -93.46 | -116.45 |
| Coding Interviews and AI | -30.20 | -43.48 | -15.63 | 14.57 |
| AI Model Preferences for Development | 56.63 | 26.92 | 10.12 | -46.51 |

The results in Table 12 show that sentiment changes did not follow the same pattern for every topic. AI Replacing Developer Jobs shifted from negative Net Sentiment in Phase 1 (−13.96) to positive values in Phases 2 (18.43) and 3 (24.47), despite increasing discussions about job displacement. This does not necessarily indicate declining career concerns, as Net Sentiment reflects the tone of discussions rather than support for job replacement. For example, a Phase 3 discussion anticipated a shift toward AI-orchestration roles while expressing concern about other jobs disappearing. The positive shift may reflect adaptation or enthusiasm toward automation, although promotional content and limitations of sentiment classification may also contribute. Meanwhile, Copilot Chat Quality Gap declined from 22.99 to −93.46, ChatGPT for Coding and Debugging fluctuated across phases, and Coding Interviews and AI remained negative. AI Model Preferences for Development remained positive but declined from 56.63 to 10.12.

Overall, these findings show that sentiment toward Generative AI changed differently across discussion topics. Although positive sentiment remained the most prevalent category, discussions about developer employment became more positive in tone, while sentiment toward AI tool quality and model preferences declined. This pattern reflects the combination of perceived benefits and concerns about GenAI in software development reported in previous studies [6], [13]. Compared with earlier research examining general public

attitudes toward ChatGPT [10], [11], and changes in developer activities over time [5], this study reveals how sentiment within specific software development topics evolved across the three observation phases.

### C. Socio-Technical Transformation of Developer Concerns

Beyond changes in topic composition and sentiment, this study examined how socio-technical concerns in developer discourse evolved across the three observation phases. The 79 canonical topics were assessed using five dimensions: Workflow Integration & Adaptation, Verification & Output Quality, Career & Employment Concerns, Role & Skill Adaptation, and Emerging Practices. Each topic was assigned to one primary dimension based on its validated label, representative keywords, and supporting documents, while topics with insufficient evidence remained unclassified. Assessment of 40 sampled topics by two coders showed high agreement (Cohen's $\kappa$ = 0.828; 87.5% agreement). The primary researcher reviewed five disagreements and also classified the remaining 39 topics. The distribution was calculated from documents assigned to classified canonical topics within each phase (P1: n = 16,677; P2: n = 17,374; P3: n = 24,620). Table 13 presents the resulting distribution of socio-technical concerns across the three observation phases.

Table 13. Socio-Technical Distribution

| Dimension | P1 | P2 | P3 |
|---|---|---|---|
| Workflow Integration & Adaptation | 47.99% | 49.34% | 26.42% |
| Verification & Output Quality | 5.17% | 2.56% | 0.74% |
| Career & Employment Concerns | 21.23% | 32.44% | 43.87% |
| Role & Skill Adaptation | 11.03% | 12.25% | 25.46% |
| Emerging Practices | 14.58% | 3.41% | 3.51% |

Table 13 shows that Workflow Integration & Adaptation dominated discussions in Phases 1 (47.99%) and 2 (49.34%), before declining to 26.42% in Phase 3. In contrast, Career & Employment Concerns increased steadily from 21.23% to 32.44% and 43.87%, becoming the dominant dimension in the final phase. Role & Skill Adaptation also increased from 11.03% to 25.46%, while Verification & Output Quality declined from 5.17% to 0.74%, and Emerging Practices decreased from 14.58% to 3.51%. The early emphasis on workflow integration is consistent with previous research on GenAI adoption in software engineering [6]. Meanwhile, the growing attention to employment and skills reflects concerns about the opportunities and risks of GenAI for developers [8], including changes in professional responsibilities [37]. Overall, these findings indicate a shift in community discussions from workflow integration toward employment, roles, and skills across the three phases. To illustrate these socio-technical concerns, Table 14 presents representative topics and their corresponding discussion contexts.

Table 14. Representative Topics and Socio-Technical Dimensions

| Phase | Representative Topic | Socio-Technical Dimension | Representative Discussion Context |
|---|---|---|---|
| P1 | ChatGPT for Coding and Debugging | Workflow Integration & Adaptation | "Scoping out projects, helping me come up with base templates for projects, doing testing, debugging" |
| P3 | AI Replacing Developer Jobs | Career & Employment Concerns | "I have replaced teams with AI. No more junior devs, and mid level gone too." |

Table 14 provides further context regarding the changes observed in Table 13. In Phase 1, the discussion highlighted the practical use of ChatGPT for project planning, testing, and debugging, reflecting an emphasis on workflow integration. In contrast, Phase 3 discussions included claims that AI would replace entry-level and mid-level developers, reflecting -growing concern about employment . Overall, these findings i+-ndicate that socio-technical concerns within developer discourse have shifted from the integration of GenAI into development workflows toward its implications for employment, professional roles, and skills. This pattern reflects a shift in the issues of concern within the developer community, rather than a verified change in actual work practices or employment conditions.

### D. From Prompt-Based Assistance to Agentic-Oriented Practices

Socio-technical changes also appear in discussions of how developers interact with Generative AI. To track the evolution of these practice orientations, 79 canonical topics were evaluated using four categories: Assistant-Oriented, Collaborative, Delegative, and Agent-Oriented. These categories represent, respectively, task-specific assistance, joint task execution, supervised delegation, and increasingly autonomous AI execution [38], [39]. Each topic was assigned a primary orientation based on validated labels, representative keywords, and supporting documents, while topics lacking sufficient evidence of developer–AI interaction were excluded. The assessment of 40 sample topics by two coders showed substantial agreement (Cohen's $\kappa$ = 0.754; 85.0% agreement). Six disagreements were reviewed by the lead researcher, who also classified the remaining 39 topics. Distributions were calculated from documents assigned to the classified practice orientations (P1: n = 12,880; P2: n = 14,725; P3: n = 18,522). Table 15 presents the results.

Table 15. Distribution of Developer–AI Practice Orientations Across Phases

| Practice Orientation | Phase 1 | Phase 2 | Phase 3 | $\Delta$ *P1 to P3* |
|---|---|---|---|---|
| Assistant-oriented | 44.82% | 43.69% | 24.77% | −20.06 |
| Collaborative | 38.53% | 20.31% | 17.95% | −20.59 |

DOI: 10.34148/teknika.v15i3.1616

| | | | | |
|---|---|---|---|---|
| Delegative | 5.61% | 11.73% | 19.44% | 13.83 |
| Agentic-oriented | 11.03% | 24.28% | 37.85% | 26.81 |

Table 15 shows that Assistant-oriented discussions dominated Phase 1 (44.82%), followed by Collaborative (38.53%), Agentic-oriented (11.03%), and Delegative (5.61%). In Phase 2, Assistant-oriented remained the largest category (43.69%), while Agentic-oriented increased to 24.28% and Collaborative declined to 20.31%. By Phase 3, Agentic-oriented became the largest category (37.85%), followed by Assistant-oriented (24.77%), Delegative (19.44%), and Collaborative (17.95%). Overall, Agentic-oriented and Delegative increased by 26.81 and 13.83 percentage points, respectively, while Assistant-oriented and Collaborative declined by 20.06 and 20.59 percentage points.

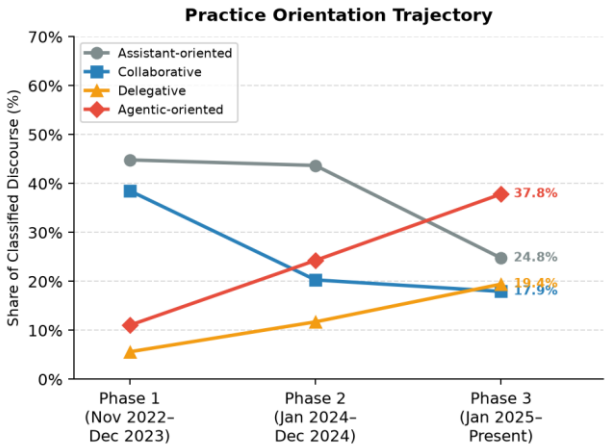



Figure 6. Shift in Developer Practice Orientations Across GenAI phases.

Figure 6 further illustrates changes in the orientation of developer–AI practices across the three observation phases. Assistant-oriented and collaborative discussions declined, while delegation-oriented and agent-oriented discussions became more common, with agent-oriented discussions becoming the dominant category in Phase 3. This pattern is further supported by discussions about integrating the Model Context Protocol (MCP) and programming agent configurations. In Phase 2, a Reddit post suggested that "Cursor with MCP in agent mode can manage code better and handle the rest," illustrating task delegation through MCP-enabled tools [40]. Similarly, a post on X highlighted the "Dispatcher-worker (planner and task worker)" approach, which reflects the coordination of autonomous agents in software development [39]. Overall, these findings indicate that developer–AI practices in online discussions have shifted from an emphasis on task-specific assistance and collaboration toward greater attention to delegated tasks and autonomous execution. However, this evolution reflects a shift in community discourse rather than a verified change in developers' actual work practices.

## IV. CONCLUSION

This study analyzed 80,570 bilingual documents from Reddit and X across three observation phases to examine changes in developer-community discourse concerning Generative AI. The identification of 79 canonical topics showed a shift in discussion focus from coding assistance and debugging toward model preferences and employment-related concerns. Although positive sentiment remained the most prevalent category, sentiment patterns varied across topics and phases. Socio-technical discussions showed increasing attention to employment and skill adaptation, while practice-oriented discussions reflected a growing emphasis on delegation and agentic execution. These findings contribute to socio-technical software engineering research by providing an integrated perspective on changes in discussion topics, sentiment, socio-technical concerns, and human–AI interaction orientations over time.

The findings may inform developers and software companies as they evaluate AI integration, human oversight, and emerging skill requirements. They may also support educators and professional training providers in identifying AI-related competencies that warrant further attention. However, the dominance of X data, potential errors in bilingual processing and automated classification, and the observational nature of social media data limit the findings' generalizability. The observed patterns therefore indicate changes in community discourse rather than confirmed changes in development practices or employment conditions. Future research should incorporate more balanced platform data, strengthen independent human validation, and combine online discussions with developer interviews or activity logs to examine whether these patterns are reflected in actual software development practices.

## REFERENCES

[1] M. Law and R. A. Varanasi, "Generative AI and Changing Work: Systematic Review of Practitioner-Led Work Transformations Through the Lens of Job Crafting," in *HCI in Business, Government and Organizations*, vol. 15804, K. L. Siau and F. F.-H. Nah, Eds., in Lecture Notes in Computer Science, vol. 15804. , Cham: Springer Nature Switzerland, 2025, pp. 131–152. doi: 10.1007/978-3-031-92823-9_10.

[2] I. Ahmed *et al.*, "Artificial Intelligence for Software Engineering: The Journey So Far and the Road Ahead," *ACM Trans. Softw. Eng. Methodol.*, vol. 34, no. 5, pp. 1–27, Jun. 2025, doi: 10.1145/3719006.

[3] J. Oertel and R. Hebig, "The Impact of Generative AI on Developer Practices, Behavior, and Software Quality," in *2025 IEEE International Conference on Software Maintenance and Evolution (ICSME)*, Auckland, New Zealand: IEEE, Sep. 2025, pp. 878–880. doi: 10.1109/ICSME64153.2025.00097.

[4] X. Hou *et al.*, "Large Language Models for Software Engineering: A Systematic Literature Review," *ACM*

*Trans. Softw. Eng. Methodol.*, vol. 33, no. 8, pp. 1–79, Nov. 2024, doi: 10.1145/3695988.

[5] A. Sergeyuk, E. Huang, D. Karaeva, A. Serova, Y. Golubev, and I. Ahmed, "Evolving with AI: A Longitudinal Analysis of Developer Logs," Mar. 30, 2026. doi: 10.1145/3744916.3787811.

[6] L. Banh, F. Holldack, and G. Strobel, "Copiloting the future: How generative AI transforms Software Engineering," *Inf. Softw. Technol.*, vol. 183, p. 107751, Jul. 2025, doi: 10.1016/j.infsof.2025.107751.

[7] Z. Feng *et al.*, "Charting Uncertain Waters: A Socio-Technical Roadmap for Sustaining Open Source Communities in the Age of GenAI," *ACM Trans. Softw. Eng. Methodol.*, vol. 35, no. 8, pp. 1–21, Aug. 2026, doi: 10.1145/3789210.

[8] J. Zacharias, A. Popova, M. V. Zahn, J. Chen, and O. Hinz, "Developers' Dilemma: Opportunities and Pitfalls of Generative AI for Software Development," *Bus. Inf. Syst. Eng.*, Mar. 2026, doi: 10.1007/s12599-026-00998-y.

[9] D. Reyes-Reina, J. M. Sanchéz-Torres, and I. M. Rueda-Cáceres, "Adoption of AI tools in software development: a systematic literature review," *Sci. Comput. Program.*, vol. 254, p. 103521, Oct. 2026, doi: 10.1016/j.scico.2026.103521.

[10] Y. Touhami Chahdi, F. M. Abbou, F. Abdi, M. Bouhadda, and L. Bouanane, "Analyzing Global Attitudes Towards ChatGPT via Ensemble Learning on X (Twitter)," *Algorithms*, vol. 18, no. 12, p. 748, Nov. 2025, doi: 10.3390/a18120748.

[11] Z. Xu, Q. Fang, Y. Huang, and M. Xie, "The public attitude towards ChatGPT on reddit: A study based on unsupervised learning from sentiment analysis and topic modeling," *PLOS ONE*, vol. 19, no. 5, p. e0302502, May 2024, doi: 10.1371/journal.pone.0302502.

[12] W. Raghupathi, J. Ren, and T. Kulkarni, "Exploring the Topics and Sentiments of AI-Related Public Opinions: An Advanced Machine Learning Text Analysis," *Information*, vol. 17, no. 2, p. 134, Feb. 2026, doi: 10.3390/info17020134.

[13] M. A. Kuhail, S. S. Mathew, A. Khalil, J. Berengueres, and S. J. H. Shah, "'Will I be replaced?' Assessing ChatGPT's effect on software development and programmer perceptions of AI tools," *Sci. Comput. Program.*, vol. 235, p. 103111, Jul. 2024, doi: 10.1016/j.scico.2024.103111.

[14] X. Chen, J. Wang, C. Gao, X. Ju, and Z. Cui, "What developers ask about openai APIs: An empirical study on stack overflow," *J. Syst. Softw.*, vol. 238, p. 112867, Aug. 2026, doi: 10.1016/j.jss.2026.112867.

[15] P. Yüce, X. Dai, R. Owens, and T. Elmas, "ChatGPT vs Teachers vs Students: Large-Scale Analysis of Generative AI Discourse in Education Communities on Reddit," May 18, 2026, *arXiv*: arXiv:2605.17712. doi: 10.48550/arXiv.2605.17712.

[16] J. Wu, V. Sivaraman, D. Kumar, J. M. Banda, and D. Sontag, "Pulse of the pandemic: Iterative topic filtering for clinical information extraction from social media," *J. Biomed. Inform.*, vol. 120, p. 103844, Aug. 2021, doi: 10.1016/j.jbi.2021.103844.

[17] D. Uniyal, M. A. Bashar, and R. Nayak, "Evaluating Cross-Lingual Classification Approaches Enabling Topic Discovery for Multilingual Social Media Data," in *Data Science and Machine Learning*, vol. 2765, Q. V. Nguyen, Y. Li, P. Kwan, Y. Zhao, Y. L. Boo, and R. Nayak, Eds., in Communications in Computer and Information Science, vol. 2765. , Singapore: Springer Nature Singapore, 2026, pp. 501–515. doi: 10.1007/978-981-95-6786-7_34.

[18] M. Lui and T. Baldwin, "langid.py: An Off-the-shelf Language Identification Tool," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju, Republic of Korea: Association for Computational Linguistics, Jul. 2012, pp. 25–30.

[19] Z. Liu, W. Lin, Y. Shi, and J. Zhao, "A Robustly Optimized BERT Pre-training Approach with Post-training," in *Chinese Computational Linguistics*, vol. 12869, S. Li, M. Sun, Y. Liu, H. Wu, L. Kang, W. Che, S. He, and G. Rao, Eds., in Lecture Notes in Computer Science, vol. 12869. , Cham: Springer International Publishing, 2021, pp. 471–484. doi: 10.1007/978-3-030-84186-7_31.

[20] D. Loureiro, F. Barbieri, L. Neves, L. Espinosa Anke, and J. Camacho-collados, "TimeLMs: Diachronic Language Models from Twitter," in *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, Dublin, Ireland: Association for Computational Linguistics, 2022, pp. 251–260. doi: 10.18653/v1/2022.acl-demo.25.

[21] G. Roccabruna, S. Azzolin, and G. Riccardi, "Multi-source Multi-domain Sentiment Analysis with BERT-based Models," in *Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022)*, Marseille, France: European Language Resources Association, Jun. 2022, pp. 581–589. [Online]. Available: https://aclanthology.org/2022.lrec-1.62

[22] J. Zhang, W. Yao, X. Chen, and L. Feng, "Transferable Post-hoc Calibration on Pretrained Transformers in Noisy Text Classification," *Proc. AAAI Conf. Artif. Intell.*, vol. 37, no. 11, pp. 13940–13948, Jun. 2023, doi: 10.1609/aaai.v37i11.26632.

[23] M. Grootendorst, "BERTopic: Neural topic modeling with a class-based TF-IDF procedure," Mar. 11, 2022, *arXiv*: arXiv:2203.05794. doi: 10.48550/arXiv.2203.05794.

[24] S. Zhang and X. Chen, "Mapping Japan's China-related twitter discourse (2010–2024) using BERTopic," *PLOS One*, vol. 21, no. 2, p. e0343085, Feb. 2026, doi: 10.1371/journal.pone.0343085.

[25] Y. Liu, D. Zhang, Q. Wan, and Z. Lin, "Innovative Mining of User Requirements Through Combined Topic Modeling and Sentiment Analysis: An

Automotive Case Study," *Sensors*, vol. 25, no. 6, p. 1731, Mar. 2025, doi: 10.3390/s25061731.

[26] L. McInnes, J. Healy, N. Saul, and L. Großberger, "UMAP: Uniform Manifold Approximation and Projection," *J. Open Source Softw.*, vol. 3, no. 29, p. 861, Sep. 2018, doi: 10.21105/joss.00861.

[27] L. McInnes and J. Healy, "Accelerated Hierarchical Density Based Clustering," in *2017 IEEE International Conference on Data Mining Workshops (ICDMW)*, New Orleans, LA: IEEE, Nov. 2017, pp. 33–42. doi: 10.1109/ICDMW.2017.12.

[28] E. Austin, S. Makwana, A. Trabelsi, C. Largeron, and O. R. Zaïane, "Uncovering Flat and Hierarchical Topics by Community Discovery on Word Co-occurrence Network," *Data Sci. Eng.*, vol. 9, no. 1, pp. 41–61, Mar. 2024, doi: 10.1007/s41019-023-00239-2.

[29] D. Stammbach, V. Zouhar, A. Hoyle, M. Sachan, and E. Ash, "Revisiting Automated Topic Model Evaluation with Large Language Models," in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing*, Singapore: Association for Computational Linguistics, 2023, pp. 9348–9357. doi: 10.18653/v1/2023.emnlp-main.581.

[30] X. Yang, H. Zhao, D. Phung, W. Buntine, and L. Du, "LLM Reading Tea Leaves: Automatically Evaluating Topic Models with Large Language Models," *Trans. Assoc. Comput. Linguist.*, vol. 13, pp. 357–375, 2025, doi: 10.1162/tacl_a_00744.

[31] L. Zheng *et al.*, "Judging LLM-as-a-Judge with MT-Bench and Chatbot Arena," in *Advances in Neural Information Processing Systems 36*, New Orleans, Louisiana, USA: Neural Information Processing Systems Foundation, Inc. (NeurIPS), 2023, pp. 46595–46623. doi: 10.52202/075280-2020.

[32] Z. Tan and J. D'Souza, "Toward purpose-oriented topic model evaluation enabled by large language models," *Int. J. Digit. Libr.*, vol. 26, no. 4, p. 23, Dec. 2025, doi: 10.1007/s00799-025-00429-5.

[33] P. Verga *et al.*, "Replacing Judges with Juries: Evaluating LLM Generations with a Panel of Diverse Models," *ArXiv Prepr. ArXiv240418796*, 2024, [Online]. Available: https://arxiv.org/abs/2404.18796

[34] J. L. Fleiss, "Measuring nominal scale agreement among many raters.," *Psychol. Bull.*, vol. 76, no. 5, pp. 378–382, Nov. 1971, doi: 10.1037/h0031619.

[35] E. Mayor, L. M. Bietti, and E. J. Canales-Rodríguez, "Text as signal. A tutorial with case studies focusing on social media (Twitter)," *Behav. Res. Methods*, vol. 55, no. 5, pp. 2595–2620, Jul. 2022, doi: 10.3758/s13428-022-01917-1.

[36] S. Abrahão, J. Grundy, M. Pezzè, M.-A. Storey, and D. A. Tamburri, "Software Engineering by and for Humans in an AI Era," *ACM Trans. Softw. Eng. Methodol.*, vol. 34, no. 5, pp. 1–46, Jun. 2025, doi: 10.1145/3715111.

[37] S. L. France, "Navigating software development in the ChatGPT and GitHub Copilot era," *Bus. Horiz.*, vol. 67, no. 5, pp. 649–661, Sep. 2024, doi: 10.1016/j.bushor.2024.05.009.

[38] D. Planötscher, X. Wang, M. Adil, P. Gregory, and S. Larimian, "Human-AI Collaboration in Software Development Activities: Perspectives of Agile Practitioners," in *Agile Processes in Software Engineering and Extreme Programming*, vol. 578, A. Goldman, S. Marczak, G. S. Tonin, D. Bassi, T. Silva Da Silva, E.-M. Schön, M. Neumann, and A. Pinna, Eds., in Lecture Notes in Business Information Processing, vol. 578. , Cham: Springer Nature Switzerland, 2026, pp. 274–289. doi: 10.1007/978-3-032-22375-3_17.

[39] J. He, C. Treude, and D. Lo, "LLM-Based Multi-Agent Systems for Software Engineering: Literature Review, Vision, and the Road Ahead," *ACM Trans. Softw. Eng. Methodol.*, vol. 34, no. 5, pp. 1–30, Jun. 2025, doi: 10.1145/3712003.

[40] Model Context Protocol, "Specification (2026-07-28)," Model Context Protocol. Accessed: Oct. 12, 2026. [Online]. Available: https://modelcontextprotocol.io/specification/2026-07-28