SOSum A Dataset of Stack Overflow Post Summaries

Bonan Kou, Yifeng Di
{koub,di5}@purdue.edu
Purdue University
West Lafayette, Indiana, USA

Muhao Chen
muhaoche@usc.edu
University of Southern California
Los Angeles, California, USA

Tianyi Zhang
tianyi@purdue.edu
Purdue University
West Lafayette, Indiana, USA

ABSTRACT

Stack Overflow (SO) is becoming an indispensable part of modern software development workflow. However, given the limited time, attention, and memory capacity of programmers, navigating SO posts and comparing different solutions is time-consuming and cumbersome. Recent research has proposed to summarize SO posts to concise text to help programmers quickly assess the relevance and quality of SO posts. Yet there is no large dataset of high-quality SO post summaries, hindering the development and evaluation of post summarization techniques. We present SOSum, a dataset of 2,278 popular SO answer posts with manually labeled summative sentences. Questions in SOSum cover 669 tags with a median view count of 253K and a median post score of 17. This dataset will foster research on sentence-level summarization of SO posts and has the potential to facilitate text summarization research on other types of textual software artifacts such as programming tutorials.

CCS CONCEPTS

• **Software and its engineering**; • **Computing methodologies** **Natural language processing**;

KEYWORDS

Stack Overflow, text summarization, data labeling

ACM Reference Format:
Bonan Kou, Yifeng Di, Muhao Chen, and Tianyi Zhang. 2022. SOSum: A Dataset of Stack Overflow Post Summaries. In *19th International Conference on Mining Software Repositories MSR '22), May 23–24, 2022, Pittsburgh, PA, USA.* ACM, New York, NY, USA, 5 pages. https://doi.org/10.1145/3524842.3528487

1 INTRODUCTION

Programmers often resort to online Q&A forums such as Stack Overflow (SO) to learn new concepts and APIs, find solutions, fix bugs, etc. [5, 9, 48, 49]. While many advanced search techniques have been proposed [10, 15, 34, 35, 52, 55], programmers still have to spend a lot of time reading, assessing, and comparing relevant Q&A posts to identify the one that best suits their context and task [11, 26, 50]. For example, a survey with 72 software developers from the industry have shown that it is cognitively demanding to sift through many online posts returned by a search engine and developers wish to get tool support for quickly navigating and assessing the relevance and quality of online posts [50].

Recently, several research efforts have been made to summarize SO posts or API documentation to facilitate users' navigation of relevant online information [18, 30, 40, 50]. In particular, Nadi and Treude experiment with four approaches to extract essential sentences from a SO post to summarize the gist of the post [30]. They run a human study with 43 developers and found that while it was promising to provide essential sentences as navigational cues, none of these approaches were sufficient to accurately identify essential sentences.

Question 133051: What is the difference between visibility:hidden and display:none?

Post 133465: They are not synonyms. <code>display:none</code> removes the element from the normal flow of the page, allowing other elements to fill in. <code>visibility:hidden</code> leaves the element in the normal flow of the page such that is still occupies space. Imagine you are in line for a ride at an amusement park and someone in the line gets so rowdy that security plucks them from the line. Everyone in line will then move forward one position to fill the now empty slot. This is like <code>display:none</code>. Contrast this with the similar situation, but that someone in front of you puts on an invisibility cloak. While viewing the line, it will look like there is an empty space, but people can't really fill that empty looking space because someone is still there. This is like <code>visibility:hidden</code>.

Figure 1: SO Post 133465 with summative sentences high lighted

The Natural Language Processing (NLP) community has made significant progress in text summarization based on recent advances in deep learning (DL) [14, 20, 21, 27, 32]. These models have achieved promising results on popular benchmarks such as GigaWord [4] and CNN/DailyMails [1]. However, these DL-based approaches are data-hungry—it requires massive parallel corpora of text documents and human-written summaries to train a model. For example, a well-known text summarization dataset, XSum [31], contains 226K BBC news articles with one-sentence summaries written by the authors and editors. Currently, there are no such high-quality datasets for Stack Overflow, which hinders the development of DL-based approaches for SO post summarization or text summarization for SE-related documents in general, such as bug reports [23, 25, 36].

To bridge the gap, we present SOSum, the first large dataset of manually curated summaries for 2,278 popular SO answer posts. Specifically, the first two authors independently labeled summative and insightful sentences from each post and then resolved disagreements to control bias and ensure labeling quality. Figure 1 shows an example SO post from SOSum with summative sentences highlighted. We also follow the question types identified in prior work [12, 42] to form a balanced dataset with 177, 182, and 147

MSR '22, May 23–24, 2022, Pittsburgh, PA, USA
2022. ACM ISBN 978-1-4503-9303-4/22/05.
https://doi.org/10.1145/3524842.3528487

answer posts respectively for three main types of questions—*how-to questions*, *conceptual questions*, and *debug-corrective questions*. These questions have an average of 362,026 view counts. Under each question, there are 5.6 answers on average. The manually curated summaries include 29 words or 1.64 sentences on average, while the average length for the posts is 74 words or 6.76 sentences. We also present a GUI tool to annotate SO posts from the SO data dump, as well as a Chrome extension for online annotation. These tools can also be used by other SE researchers to construct larger datasets on SO posts and other types of SE-related documents such as bug reports in future research. Both our dataset and our data labeling tools are publicly available on GitHub [2].

2 RELATED WORK

Our work is motivated by several empirical studies on searching and navigating relevant posts on Stack Overflow [11, 26, 49, 50]. Xia et al. analyzed search queries from 60 developers and found that finding explanations for unknown concepts, learning how to implement a task or use an API, and finding solutions to fix bugs are the most frequent search tasks [49]. Xu et al. conducted a formative study with 72 developers and found that information overload (e.g., too many relevant SO posts, too much text in a post) is the main challenge of identifying useful information from online Q&A forums [50]. In an exploratory study with 50 novice programmers, Chatterjee et al. found that 69% of participants considered too much text containing unnecessary details as the main reason that slows down the navigation of SO posts [11].

Recently, several techniques have been proposed to summarize SO posts to facilitate SO post navigation and information seeking [30, 40, 46, 50]. The most related work is AnswerBot, which extracts summative paragraphs from SO posts based on various features such as information entropy and paragraph position [50]. Compared with AnswerBot, our work focuses on more fine-grained post summarization at the sentence level. CraSolver generates a summary of bug solutions by selecting important paragraphs from relevant SO posts based on a multi-factor ranking mechanism [46]. Nadi and Treude experimented with four different information retrieval or pattern-matching approaches to select essential sentences from SO posts to help programmers navigate SO posts [30]. Through a survey with 43 developers, they found that, while participants would indeed like to get navigation support on Stack Overflow, none of the four approaches were sufficient to provide such support.

Recent advancements in NLP have made it possible to generate concise summaries of text documents, such as news articles, using deep neural networks [14, 27, 29, 31, 32, 45]. For example, Liu et al. present a news article summarization approach by fine-tuning a language model [13] with over 300K unique CNN and Daily Mail news articles along with their news summaries written by article authors [27]. Narayan et al. present an approach that summarizes a single document to a one-liner using a convolutional neural network trained on the XSum corpus [31]. Later, Narayan et al. present another approach in which the task of text summarization is framed as a ranking problem and solved by globally optimizing the ROGUE metric in training [32]. Dong et al. frame text summarization as a contextual bandit problem, in which each document is considered as a context and each combination of selected sentences is an action to take [14]. While these models perform well on news articles, reusing these models in another domain, such as SO posts, requires fine-tuning with large-scale parallel corpora due to vocabulary and language norm shift. To the best of our knowledge, there are no such corpora with SO posts and post summaries.

In addition to post summarization, there are many other approaches for supporting information seeking on Stack Overflow, e.g., integrating SO into an IDE [33, 34], query reformulation for better search results [35, 52], identifying insightful sentences about API usage [22, 41, 43], summarizing API opinions [24, 44], detecting API misuses [38, 53] or deprecated APIs [54] in SO posts, etc. Due to the page limit, we will not elaborate here.

3 DATASET CONSTRUCTION

This section describes the process of curating the dataset of 2278 popular SO answer posts and their post summaries.

3.1 SO Post Pre processing

We first extracted SO questions from the Stack Overflow data dump [3] and ranked them based on their view counts. We chose view counts since we wanted to focus on popular SO posts first. Then, the first author manually inspected the frequently viewed questions and selected three types of commonly asked questions—*how-to* questions, *conceptual* questions, and *debug-corrective* questions based on the taxonomy from prior work [12, 42]. Eventually, the first author selected 177 *how-to*, 182 *conceptual*, and 147 *debug-corrective* questions to form a balanced dataset. For each question post, we filtered out its answer posts with negative scores since they may not be valid answers. We also removed answer posts that only contain code snippets. For the remaining 2,283 answer posts, we extracted their content and other metadata such as view counts and post scores from the SO data dump. Since the content of each post was stored as plain HTML in the data dump, we removed the HTML tags and used the NLTK package [7] to break natural language descriptions into sentences. Pre-processed question posts and answer posts were stored in CSV.

3.2 SO Post Labeling Tools

We developed a graphical user interface (GUI) to facilitate labeling summative sentences in SO posts, as shown in Figure 2. A user can load the pre-processed SO posts into this tool by clicking the "Select File" button. Then the data labeling tool will show the current labeling progress, including the number of answer posts in total, the index of the current post, and the number of labeled posts. It renders one answer post and its corresponding question at a time. The content of the rendered answer post is broken into sentences for the ease of labeling. Some metadata such as SO tags, post score, and post URL are also rendered. After reading the question title and content, a data labeler can navigate through the sentences and label one or more sentences as a summary of the answer post. They can click "Next" to move on to the next post or "Previous" to change the labeling of previous posts.

As many new SO posts are posted every day, one may also be interested in labeling SO posts from the website rather than from the data dump. Therefore, we also developed a Chrome extension

Table 1: Bookmarks and view count for the 506 SO questions

	min	max	average	median
Bookmarks	1	5,385	10	75
View count	115,672	632,137	362,026	253,527

Table 2: Statistics of 2283 answer posts and their summaries.

	min	max	average	median
Post body length (words)	1	1,320	74	50
Summary length (words)	0	346	29	21
Post Score	0	7,521	125	17

for labeling SO posts in a web browser. A user can select one or more sentences from a post, right click, and choose "Mark as Summative Sentence" to label them as summative sentences. Once the labeling is done, the user can press "CTRL+ENTER" to download the post content as well as the labeled summative sentences into a CSV file. A video demo of this extension and both tools have been made publicly available on GitHub to support Open Science [2].

3.3 SO Post Labeling Pipeline

We followed the common standard of NLP data labeling process to label SO posts [8, 39, 51]. The first two authors first independently labeled summative sentences in 464 answer posts from the first 100 questions in the dataset. Then, the authors compared their labeling results and resolved the inconsistencies. The initial Cohen's kappa score is 0.41. After discussion, a set of labeling rules were summarized to mitigate bias and ensure labeling consistency. Based on these labeling rules, the first two authors labeled 526 posts from the second batch of 100 questions in the dataset. The authors met again and compared their labeling results. The kappa score of the second-round labeling is increased to 0.67, which indicates a moderate agreement under the guidance of the labeling rules [28]. The authors resolved some inconsistencies and further refined the labeling rules. Finally, they continued to label the rest 1288 answer posts and resolved inconsistencies. The final SOSum dataset only includes summative sentences that both authors agree upon. This whole labeling process takes about 175 man-hours.

The final set of labeling rules are summarized below.

Prefer sentences that directly answer a question. For example, in Figure 1, sentence 1-3 all contribute to the post on a high level but since the question asks for the difference between visibility:hidden and display:none, we select sentence 2 and 3.

Prefer topic sentences over illustrative descriptions. Sentences such as "for example,..." are not preferred, while sentences that describe a high-level concept or idea are preferred.

Prefer instructions over explanations Sentences with clear instructions on how to complete a programming task or fix a bug, if any, are selected. Sentences explaining these instructions are not.

Select a complete list of items or steps. If one sentence in a bullet list or procedure (e.g. *"First, try A. Then, do B."*) is selected, the corresponding sentences for the following items or steps should also be selected to ensure completeness and consistency.

Prefer concise sentences over verbose sentences. If there are two summative sentences, we always pick the shorter one.

Table 3: 10 most popular tags in the 506 SO questions

Tag	Question count	Tag	Question count
Java	96	Sql-server	29
Javascript	52	C++	24
C	48	JQuery	21
Python	38	PHP	21
Sql	31	.Net	19

Table 4: Data Fields in SOSum

Field	Description
Question Id	Post Id of the SO question
Question Type	1 for *conceptual* questions, 2 for *how-to* questions, 3 for *debug-corrective* questions
Question Title	Question title as a string
Question Body	A list of sentences from the question post content
Tags	SO tags associated with a question
Answer Posts	A list of post ids separated by comma
Answer Id	Post Id of a SO answer post
Answer Body	A list of sentences from the post content
Summary	Summative sentences from the post content

4 DATASET DESCRIPTION

Our final dataset includes 2,278 answer posts and their summaries. These posts are from 506 frequently viewed SO questions. The median of bookmarks and view counts of these questions are 10 and 36K respectively, as shown in Table 1. These questions also cover diverse topics in different programming languages and libraries. Table 3 shows the 10 most popular tags out of a total number of 669 unique tags associated with these questions.

Table 2 describes the statistics of the 2,278 answer posts in SOSum. The median number of words in these answer posts and their summaries is 50 and 21 respectively. The median score (i.e., upvotes minus downvotes from other SO users) is 17. Figure 3 shows the distribution of post length and summary length in terms of words. 90% of posts have less than 272 words, while 90% of the summaries are comprised of 65 words or less in total.

Table 4 describes the data fields in our dataset. We store the question post data and the answer post data in two separate CSV files. These CSV files can be directly loaded into the desktop data labeling tool described in Section 3.2.

5 USE CASES

This section describes how SOSum can be used to develop SO post summarization techniques as well as three other potential use cases.

Sentence-level SO post summarization. Despite rapid progress in NLP, sentence-level text summarization approaches have been absent in the Stack Overflow domain due to a lack of parallel corpora required for training or fine-tuning existing models. With SOSum, SE researchers can fine-tune pre-trained NLP models in the general NLP domain for summarizing SO posts. Some advanced NLP models SE researchers can fine-tune include BERT, Sentence-bert, and SpanBERT [19, 27, 37]. For example, one can fine-tune BERT [13] using SOSum as a sentence-level Conditional Random Field (CRF) model [27] to predict whether a sentence in a SO post is a summative sentence.

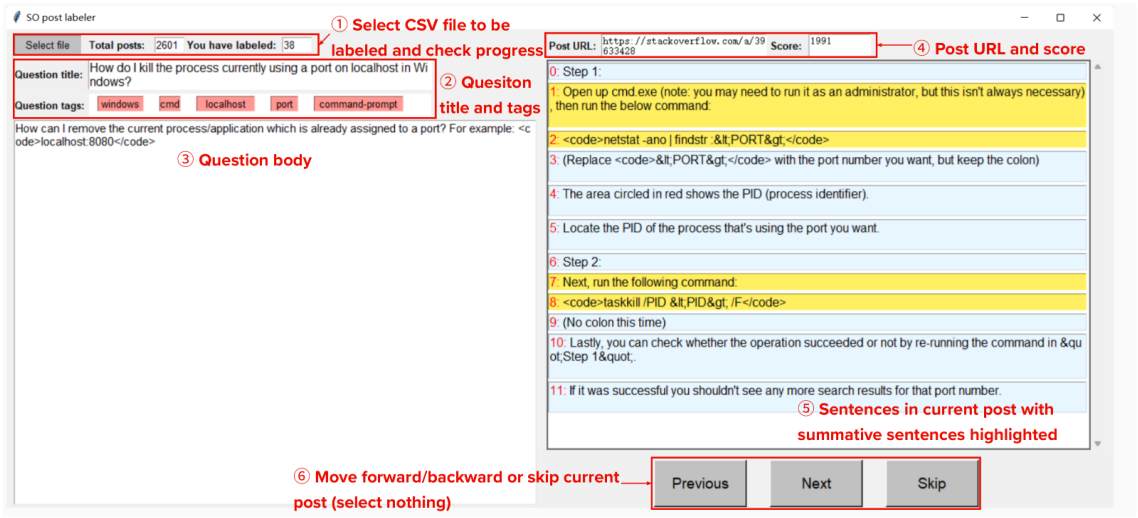

Figure 2: A screenshot of the SO post labeling user interface

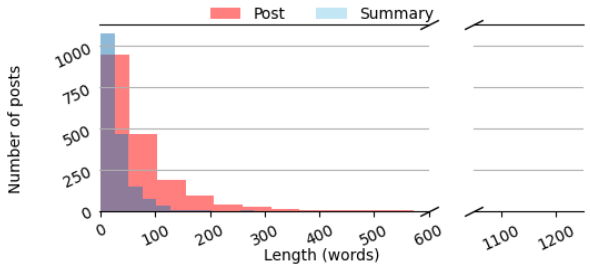

Figure 3: Frequency of SO posts of different lengths

Question answering QA). A QA system can automatically generate an answer to a question expressed in natural language [16]. Developing a QA system often requires a large corpus of question-answer pairs. Since SOSum includes pairs of question titles and answer summaries, it can be naturally used to build a QA system for answering programming questions. Existing systems for answering programming questions mostly rely on rule-based pattern matching or unsupervised methods [6, 15, 40, 50]. SOSum offers a large training dataset that can be used to fine-tune more advanced DNN-based QA models [16, 17, 47] to answer programming questions.

Summarizing other types of text documents in SE. The lack of training data with high-quality labels hinders the development of NLP models for other types of SE documents, such as programming tutorials and bug reports. Since these documents often share similar language norms and vocabularies, it is possible to perform transfer learning. Therefore, SOSum also creates new research opportunities to introduce data-hungry DL models for other types of SE documents.

Benchmarking. SOSum can be used to benchmark existing text summarization tools for Stack Overflow [30, 40, 46, 50]. Researchers have been relying on a small benchmark or user studies to evaluate existing tools. Compared with using SOSum as a benchmark, user studies are inevitably limited in scale and also take a long time to complete. SOSum on the other hand provides an easier and faster way for researchers to test the accuracy of their tools on a large, human-validated dataset.

6 THREATS TO VALIDITY

One potential threat to validity comes from the data labeling process, since human labelers may introduce errors or biases. Furthermore, different programmers may prefer different summative sentences. To mitigate this threat, the first two authors conducted three rounds of independent labeling, standardized the labeling rules, and discussed extensively to resolve inconsistencies. Furthermore, our final dataset only includes summative sentences that both data labelers agree upon. Another threat is that SOSum only includes 2,278 pairs of answer posts and their summaries. While it is much bigger than existing benchmarks used in prior work [30, 40, 46, 50], it may not be sufficient to train a large NLP model from scratch. To address this issue, we have released a GUI tool and a Chrome extension for labeling new SO posts. We wish these tools would also encourage other SE researchers to construct large-scale, high-quality training datasets for Stack Overflow and other types of SE documents.

7 CONCLUSION

Previous studies have shown that navigating SO posts is time-consuming and programmers wish to get tool support for quickly understanding and assessing SO posts. We propose SOSum, a dataset of 2,278 popular SO posts with manually labeled summaries to foster research on sentence-level SO post summarization. In addition to the dataset itself, we also present two SO post labeling tools for offline and online labeling. By making the dataset and data labeling tools publicly available, we wish to encourage the construction of more large-scale datasets with high-quality labels for Stack Overflow and other types of SE documents. In the future, we plan to add more labeled data to SOSum, conduct a systematic evaluation of existing SO post summarization tools using SOSum, and fine-tune advanced NLP models for SO post summarization.

REFERENCES

[1] 2022. Document Summarization on CNN/Daily Mail. https://paperswithcode.com/sota/document-summarization-on-cnn-daily-mail. Accessed: 2022-3-29.

[2] 2022. The GitHub repository of SOSum and SO post labeling tools. https://github.com/BonanKou/SOSum-A-Dataset-of-Extractive-Summaries-of-Stack-Overflow-Posts-and-labeling-tools.

[3] 2022. Stack Exchange Data Dump. https://archive.org/details/stackexchange.

[4] 2022. Text Summarization on GigaWord. https://paperswithcode.com/sota/text-summarization-on-gigaword. Accessed: 2022-3-29.

[5] Rabe Abdalkareem, Emad Shihab, et al. 2017. What do developers use the crowd for? a study using stack overflow. *IEEE Software* 34, 2 (2017), 53–60.

[6] Ahmad Abdellatif, Khaled Badran, et al. 2020. MSRBot: Using bots to answer questions from software repositories. *Empirical Software Engineering* 25, 3 (2020), 1834–1863.

[7] Steven Bird, Ewan Klein, et al. 2009. *Natural language processing with Python: analyzing text with the natural language toolkit.* " O'Reilly Media, Inc.".

[8] ElMehdi Boujou, Hamza Chataoui, Abdellah El Mekki, Saad Benjelloun, Ikram Chairi, and Ismail Berrada. 2021. An open access NLP dataset for Arabic dialects: Data collection, labeling, and model construction. *arXiv preprint arXiv:2102.11000* (2021).

[9] Joel Brandt, Philip J Guo, et al. 2009. Two studies of opportunistic programming: interleaving web foraging, learning, and writing code. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*. 1589–1598.

[10] Kaibo Cao, Chunyang Chen, et al. 2021. Automated Query Reformulation for Efficient Search based on Query Logs From Stack Overflow. In *2021 IEEE/ACM 43rd International Conference on Software Engineering ICSE)*. IEEE, 1273–1285.

[11] Preetha Chatterjee, Minji Kong, et al. 2020. Finding help with programming errors: An exploratory study of novice software engineers' focus in stack overflow posts. *Journal of Systems and Software* 159 (2020), 110454.

[12] Lucas BL De Souza, Eduardo C Campos, et al. 2014. Ranking crowd knowledge to assist software development. In *Proceedings of the 22nd International Conference on Program Comprehension*. 72–82.

[13] Jacob Devlin, Ming-Wei Chang, et al. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).

[14] Yue Dong, Yikang Shen, et al. 2018. Banditsum: Extractive summarization as a contextual bandit. *arXiv preprint arXiv:1809.09672* (2018).

[15] Swapna Gottipati, David Lo, et al. 2011. Finding relevant answers in software forums. In *2011 26th IEEE/ACM International Conference on Automated Software Engineering ASE 2011)*. IEEE, 323–332.

[16] Karl Moritz Hermann, Tomas Kocisky, et al. 2015. Teaching machines to read and comprehend. *Advances in neural information processing systems* 28 (2015), 1693–1701.

[17] Hsin-Yuan Huang, Chenguang Zhu, et al. 2017. Fusionnet: Fusing via fully-aware attention with application to machine comprehension. *arXiv preprint arXiv:1711.07341* (2017).

[18] Qiao Huang, Xin Xia, et al. 2018. API method recommendation without worrying about the task-API knowledge gap. In *2018 33rd IEEE/ACM International Conference on Automated Software Engineering ASE)*. IEEE, 293–304.

[19] Mandar Joshi, Danqi Chen, et al. 2020. Spanbert: Improving pre-training by representing and predicting spans. *Transactions of the Association for Computational Linguistics* 8 (2020), 64–77.

[20] D. Khashabi, S. Min, et al. 2020. UnifiedQA: Crossing Format Boundaries With a Single QA System. *EMNLP - findings* (2020).

[21] Mike Lewis, Yinhan Liu, et al. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461* (2019).

[22] Hongwei Li, Sirui Li, et al. 2018. Improving api caveats accessibility by mining api caveats knowledge graph. In *2018 IEEE International Conference on Software Maintenance and Evolution ICSME)*. IEEE, 183–193.

[23] Xiaochen Li, He Jiang, et al. 2018. Unsupervised Deep Bug Report Summarization. In *2018 IEEE/ACM 26th International Conference on Program Comprehension ICPC)*. 144–14411.

[24] Bin Lin, Fiorella Zampetti, et al. 2019. Pattern-based mining of opinions in q&a websites. In *2019 IEEE/ACM 41st International Conference on Software Engineering ICSE)*. IEEE, 548–559.

[25] Haoran Liu, Yue Yu, et al. 2020. *BugSum: Deep Context Understanding for Bug Report Summarization.* Association for Computing Machinery, New York, NY, USA, 94–105. https://doi.org/10.1145/3387904.3389272

[26] Jiakun Liu, Sebastian Baltes, et al. 2021. Characterizing Search Activities on Stack Overflow. (2021).

[27] Yang Liu. 2019. Fine-tune BERT for extractive summarization. *arXiv preprint arXiv:1903.10318* (2019).

[28] Mary L McHugh. 2012. Interrater reliability: the kappa statistic. *Biochemia medica* 22, 3 (2012), 276–282.

[29] Derek Miller. 2019. Leveraging BERT for extractive text summarization on lectures. *arXiv preprint arXiv:1906.04165* (2019).

[30] Sarah Nadi and Christoph Treude. 2020. Essential sentences for navigating stack overflow answers. In *2020 IEEE 27th International Conference on Software Analysis, Evolution and Reengineering SANER)*. IEEE, 229–239.

[31] Shashi Narayan, Shay B Cohen, et al. 2018. Don't Give Me the Details, Just the Summary *Topic-aware Convolutional Neural Networks for Extreme Summarization. In* (2018).

[32] Shashi Narayan, Shay B Cohen, et al. 2018. Ranking sentences for extractive summarization with reinforcement learning. *arXiv preprint arXiv:1802.08636* (2018).

[33] Luca Ponzanelli, Alberto Bacchelli, et al. 2013. Seahawk: Stack overflow in the ide. In *2013 35th International Conference on Software Engineering ICSE)*. IEEE, 1295–1298.

[34] Luca Ponzanelli, Gabriele Bavota, et al. 2014. Mining stackoverflow to turn the ide into a self-confident programming prompter. In *Proceedings of the 11th Working Conference on Mining Software Repositories*. 102–111.

[35] Mohammad Masudur Rahman and Chanchal Roy. 2018. Effective reformulation of query for code search using crowdsourced knowledge and extra-large data analytics. In *2018 IEEE International Conference on Software Maintenance and Evolution ICSME)*. IEEE, 473–484.

[36] Sarah Rastkar, Gail C. Murphy, et al. 2014. Automatic Summarization of Bug Reports. *IEEE Transactions on Software Engineering* 40, 4 (2014), 366–380. https://doi.org/10.1109/TSE.2013.2297712

[37] Nils Reimers and Iryna Gurevych. 2019. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084* (2019).

[38] Anastasia Reinhardt, Tianyi Zhang, et al. 2018. Augmenting stack overflow with API usage patterns mined from GitHub. In *Proceedings of the 2018 26th ACM Joint Meeting on European Software Engineering Conference and Symposium on the Foundations of Software Engineering*. 880–883.

[39] Filipe Rodrigues, Francisco Pereira, and Bernardete Ribeiro. 2014. Sequence labeling with multiple annotators. *Machine learning* 95, 2 (2014), 165–181.

[40] Rodrigo FG Silva, Chanchal K Roy, et al. 2019. Recommending comprehensive solutions for programming tasks by mining crowd knowledge. In *2019 IEEE/ACM 27th International Conference on Program Comprehension ICPC)*. IEEE, 358–368.

[41] Siddharth Subramanian, Laura Inozemtseva, et al. 2014. Live API documentation. In *Proceedings of the 36th International Conference on Software Engineering*. 643–652.

[42] Christoph Treude, Ohad Barzilay, et al. 2011. How do programmers ask and answer questions on the web?(nier track). In *Proceedings of the 33rd international conference on software engineering*. 804–807.

[43] Christoph Treude and Martin P Robillard. 2016. Augmenting api documentation with insights from stack overflow. In *2016 IEEE/ACM 38th International Conference on Software Engineering ICSE)*. IEEE, 392–403.

[44] Gias Uddin and Foutse Khomh. 2017. Opiner: an opinion search and summarization engine for APIs. In *2017 32nd IEEE/ACM International Conference on Automated Software Engineering ASE)*. IEEE, 978–983.

[45] Sukriti Verma and Vagisha Nidhi. 2017. Extractive summarization using deep learning. *arXiv preprint arXiv:1708.04439* (2017).

[46] Haoye Wang, Xin Xia, et al. 2021. Automatic Solution Summarization for Crash Bugs. In *2021 IEEE/ACM 43rd International Conference on Software Engineering ICSE)*. IEEE, 1286–1297.

[47] Wenhui Wang, Nan Yang, et al. 2017. Gated self-matching networks for reading comprehension and question answering. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics Volume 1: Long Papers)*. 189–198.

[48] Yuhao Wu, Shaowei Wang, et al. 2019. How do developers utilize source code from stack overflow? *Empirical Software Engineering* 24, 2 (2019), 637–673.

[49] Xin Xia, Lingfeng Bao, et al. 2017. What do developers search for on the web? *Empirical Software Engineering* 22, 6 (2017), 3149–3185.

[50] Bowen Xu, Zhenchang Xing, et al. 2017. AnswerBot: Automated generation of answer summary to developers' technical questions. In *2017 32nd IEEE/ACM International Conference on Automated Software Engineering ASE)*. IEEE, 706–716.

[51] Jinghang Xu, Wanli Zuo, Shining Liang, and Xianglin Zuo. 2020. A review of dataset and labeling methods for causality extraction. In *Proceedings of the 28th International Conference on Computational Linguistics*. 1519–1531.

[52] Neng Zhang, Qiao Huang, et al. 2020. Chatbot4qr: Interactive query refinement for technical question retrieval. *IEEE Transactions on Software Engineering* (2020).

[53] Tianyi Zhang, Ganesha Upadhyaya, et al. 2018. Are code examples on an online Q&A forum reliable?: a study of API misuse on stack overflow. In *2018 IEEE/ACM 40th International Conference on Software Engineering ICSE)*. IEEE, 886–896.

[54] Jing Zhou and Robert J Walker. 2016. API deprecation: a retrospective analysis and detection method for code examples on the web. In *Proceedings of the 2016 24th ACM SIGSOFT International Symposium on Foundations of Software Engineering*. 266–277.

[55] Yanzhen Zou, Ting Ye, et al. 2015. Learning to rank for question-oriented software text retrieval (t). In *2015 30th IEEE/ACM International Conference on Automated Software Engineering ASE)*. IEEE, 1–11.