2019 IEEE/ACM 16th International Conference on Mining Software Repositories (MSR)

Python Coding Style Compliance on Stack Overflow

Nikolaos Bafatakis, Niels Boecker, Wenjie Boon, Martin Cabello Salazar, Jens Krinke, Gazi Oznacar, Robert White
University College London, UK

***Abstract*—Software developers all over the world use Stack Overflow (SO) to interact and exchange code snippets. Research also uses SO to harvest code snippets for use with recommendation systems. However, previous work has shown that code on SO may have quality issues, such as security or license problems.**

We analyse Python code on SO to determine its coding style compliance. From 1,962,535 code snippets tagged with 'python', we extracted 407,097 snippets of at least 6 statements of Python code. Surprisingly, 93.87% of the extracted snippets contain style violations, with an average of 0.7 violations per statement and a huge number of snippets with a considerably higher ratio. Researchers and developers should, therefore, be aware that code snippets on SO may not representative of good coding style.

Furthermore, while user reputation seems to be unrelated to coding style compliance, for posts with vote scores in the range between -10 and 20, we found a strong correlation ($r = -0.87, p < 10^{-7}$) between the vote score a post received and the average number of violations per statement for snippets in such posts.

***Index Terms*—Stack Overflow, Style Guides, Code Style, Coding Style, Coding Conventions, SOTorrent, Python**

I. Introduction

Stack Overflow (SO) is a community-based question answering (CQA) website where programmers seek help from their peers. Content therein is regulated via a community voting mechanism, where good content gets voted up and bad content gets voted down, giving posts a vote score. Through such interactions, users are given reputation scores, which are constantly changing based on their behaviour. Code-related discussions are the most common, as 64% of posts contain code snippets [1]. However, such snippets may have quality issues, such as security problems [2] or potential software licensing problems [3], [4].

It is commonly said that code is read more often than it is written [5], [6], [7]. This is especially true for a site like SO, where code within questions and answers is constantly referenced by other users. But code can be written in many ways, leading to confusion or ambiguity. To mitigate this, coding style guides have been established and are commonly used to ensure code is represented consistently and clearly. However, their compliance and effectiveness in CQA sites are scarcely investigated. Though there have been previous studies involving SO code snippets [8], [9], [10], [11], there have not been any studies focusing primarily on coding style compliance and its correlation with the popularity of users and posts.

In this study, we assess SOTorrent [12], a dataset containing historical data from SO, to fill this research gap. Our results show that 93.87% of analysed Python code snippets contain coding style violations, and while there is a correlation between coding style compliance and post score, reputation and coding style compliance seem to be uncorrelated.

II. Background

Code snippets in questions and answers on SO need to be readable and understandable to serve their purpose. However, a study [11] showed that, over a sample of Java snippets, less than half of the snippets were considered to be self-explanatory. Another study for Java showed that the most important code metrics related to question quality are those most important for general code readability [10].

Style guides are guidelines that dictate how code should be represented. From syntax elements to naming conventions, they ensure that code is as readable as possible for programmers [13], [14], eventually facilitating maintenance [15], [6]. While code convention adherence is perceived as important by practitioners [16], it is difficult for developers to comply [15], [17]. To simplify and automate the process, many coding style checkers have been created to validate the compliance of a codebase with a language's style guide [18], [19], [20].

There is no universal set of rules that can be applied to all programming languages and different languages have their own coding style definitions [21], [22], [23]. Certain languages even induced multiple guides created by different companies or groups; JavaScript has at least 5 different style guides [24]. Others, such as Python, have a general widely accepted coding style that is typically adopted with only slight variations [25].

III. Research Questions

In order to determine if coding style guides are followed on SO and if compliance and popularity are correlated, we analysed Python code snippets to answer the following questions:

- **RQ1** Do code snippets generally tend to comply with coding style guides?
- **RQ2** Which coding style rules are broken most frequently?
- **RQ3** Do posts complying with coding style guides receive more favourable votes?
- **RQ4** Do high reputation users tend to comply with coding style guides more than low reputation users?

IV. Experimental Design

A. Selection of Programming Languages

We selected Python to be the focus of this study as it is one of the most popular languages on SO and was the highest trending language at the time of writing [26]. Python's flexible language structure allows style check tools to work with partial and incomplete code, which is commonly seen on SO [9]. Much previous work investigating code snippets on SO focussed on parseable Java snippets, however, only 3.89% of Java snippets have been found to be parseable while 76% of Python snippets were parseable [27].

2574-3864/19/$31.00 ©2019 IEEE
DOI 10.1109/MSR.2019.00042

B. The SOTorrent Dataset

SOTorrent [12] is an open dataset based on data from the official SO data dump. The dataset covers all SO posts and user information since the first post in July 2008. Unlike in the raw data dump, posts are broken down to individual paragraphs ("blocks"), which are classified as either code or text. The dataset also contains a complete version history of posts and blocks as well as other information like user reputation and links to GitHub repositories with links to SO.

C. Style Checking

We evaluated two style checking tools that are commonly used for Python: Pylint and Flake8. Both cover the widely accepted PEP8 style guide [23] and flag violations for fragmented code snippets. We selected Pylint because it is highly configurable and widely accepted by the Python community [18]. We configured the tool to exclude the following rules as they are not important in the context of partial code snippets:

- import-error (imports are not important)
- unused-wildcard-import (many incomplete snippets)
- missing-docstring (docstrings are not necessary)
- undefined-variable (fake variables are commonly used)
- missing-final-newline (are often skipped in short snippets)

Also, we allowed any naming style for constants, as snippets are usually in global scope and all variables in global scope are regarded as constants. Additionally, we allowed any naming style for modules as the snippets do not have a file name.

D. Measuring Coding Style Compliance

We extracted all code snippets from posts tagged with 'python' in the most recent version of the SOTorrent dataset, version 2018-12-09, yielding 1,962,535 rows, each one representing a supposed Python code snippet. Next, we ran each snippet through several processing steps, eventually either filtering it out or obtaining the number of each type of coding style violation within the snippet. Given that the snippets include artefacts of the SOTorrent extraction (e.g., escaped newline characters), we first cleansed them to reconstruct the exact code the user wrote. Then, we filtered all snippets that have fewer than 6 lines (a size commonly used [28]) or do not contain any of four basic tokens usually found in Python syntax ('`print`', '`import`', '`(`', '`=`'). Afterwards, we ran the snippets through Pylint to check for violations in both Python 2 and Python 3 runtime environments and the result containing fewer violations was selected. Python 2 and 3 differ significantly, e.g., `print` arguments must be in parentheses in Python 3 but such parentheses generate a style violation in Python 2. Finally, code snippets that could not be processed in either environment were assumed to be plain text or non-Python code. For all remaining snippets, a report of violated rules was generated and is made publicly available [29]. Also, the metadata provided by Pylint allowed us to then filter out snippets with fewer than 6 statements. Table I shows the breakdown of the number of processed results and filtered data. In the end, 407,097 (21%) of the extracted snippets fulfil the above constraints and were used as the data set for further analysis.

Table I
OVERVIEW OF THE NUMBER OF CODE SNIPPETS IN THE VARIOUS PROCESSING STEPS

Filter	Snippets
None	1,962,535
≥ 6 lines & contains '`print`', '`import`', '`(`' or '`=`'	863,122
Processable by pylint	462,393
≥ 6 statements	407,097

V. RESULTS

A. RQ1: General Compliance

After filtering and processing the data as described above, 407,097 snippets remained. Of these, we were surprised to find only 6.91% (24,972) do not violate any coding style rules, while a majority (93.87% – 382,125) of snippets contain one or more violations. On average, there are 0.7 violations per statement. This result shows that the Python code on SO does not tend to comply with coding style conventions. In fact, there even is a huge number of snippets with a considerably higher ratio. Researchers and developers should, therefore, be aware that Python code snippets on SO are *not representative of good coding style* and may not be suitable for learning approaches like Naturalize [17] or Butler et al.'s work on mining Java class name conventions [30].

B. RQ2: Top Violations

In all code snippets for which Pylint was able to determine the coding style violations, we found 5,076,647 rule violations in total. The rule most commonly broken (33.4% of all violations) is *bad-whitespace*: "used when a wrong number of spaces is used around an operator, bracket, or comma, or before a block opener colon" [31]. The 10 most commonly violated rules make up for 85.1% of all detected violations.

We classified the 5,076,647 detected violations according to the four categories defined in the Pylint documentation [31]. This indicated that most violations fall into the *Convention* category (3,378,361), followed by *Warning* (1,443,576) and *Refactor* (201,006), with *Error* being the least common (53,677). Table II shows, for each category, the 5 most detected violations. The 'R' column shows the overall rank of the violation.

C. RQ3: Compliance and Votes

This RQ aims at analysing whether the post score, i.e. the aggregated upvotes and downvotes a post received from the SO community, is correlated with the number of coding style violations per statement, referred to as the violation ratio.

Figure 1 shows the relationship between post score and violation ratio when grouped by discrete post score values. Essentially this shows a Zipfian distribution: There are many posts with low scores and a low violation ratio, a small number of posts with high scores and low violation ratio, a small number of posts with high violation ratio and low score, but no posts with both a high violation ratio and high score.

When considering the post score values closer to zero and disregarding the rare higher ones, the outliers with extreme violation ratios strongly distort the mean. E.g., the post with

Table II
OVERVIEW OF THE TOP-5 VIOLATIONS FOR EACH CATEGORY

Error	Number	R	Warning	Number	R	Refactor	Number	R	Convention	Number	R
no-member	15,381	24	bad-indentation	446,575	4	too-few-public-methods	112,555	9	bad-whitespace	1,697,550	1
used-before-assignment	8,363	36	mixed-indentation	419,225	5	no-self-use	30,608	15	invalid-name	636,406	2
return-outside-function	4,745	44	unused-import	139,810	8	no-else-return	15,203	25	trailing-whitespace	506,414	3
relative-beyond-top-level	4,205	47	redefined-outer-name	70,571	11	inconsistent-return-statements	9524	33	bad-continuation	145,572	6
no-value-for-parameter	2,656	53	unused-variable	59,926	12	too-many-arguments	5846	40	line-too-long	144,638	7

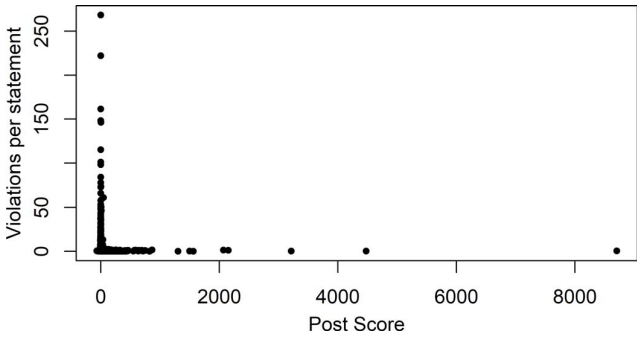

Figure 1. Scatter plot showing violations per statement over post score

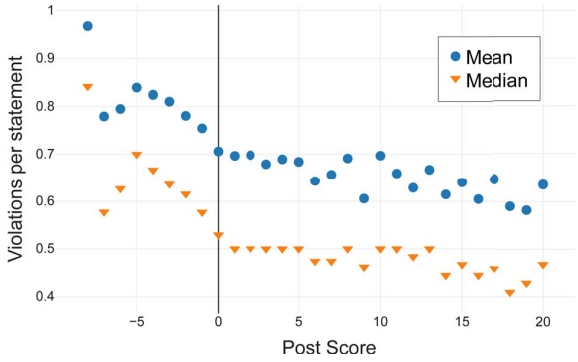

Figure 2. Mean and median of violations per statement grouped by post score

ID 27762547 yields 1,150 bad-whitespace errors because of an inline array with repeated whitespace violations. As there are only 15 posts with a post score of 48, the single post leads to a significant spike of the violation ratio for this post score value. Hence, when the correlation is examined over the whole dataset, it is dominated by noise and close to non-existent.

Consequently, in order to make meaningful observations, the complete set of values must be filtered and smoothed. We focus on the range of posts that have received a post score within the range of -10 and +20, as the vast majority of posts (99.3%) fall into this range, making a correlation most meaningful here. Additionally, we discard any posts that fall into a post score group with fewer than 50 posts, as we found this threshold to minimise the distortion through outliers considerably, while not discarding too much relevant data. Consequently, the effective range becomes [-8, +20]. With these restrictions applied, we then analyse the mean and median values of violations per statement for each group of posts sharing the same post score.

As Figure 2 illustrates, both the mean and median values indeed exhibit a strong correlation with the post score: The mean values of the violation ratio have a Pearson correlation coefficient of $r = -0.87, p < 10^{-7}$. Considering the median values instead, the result is very similar: $r = -0.82, p < 10^{-7}$. This result is highly significant, as it implies that posts containing coding style compliant snippets get more favourable votes from the SO community, and vice versa, code snippets with higher violation ratios are voted worse.

When the upper border is extended beyond post score values of 20, the correlation decreases, as can be seen in Table III. Shifting the lower border is not helpful, as most values would be filtered out due to low numbers of posts per post score. Given that the original range comprises the bulk of the posts, the modified ranges are less relevant. Seeing the correlation decrease therefore even enhances our confidence in the finding that the community-voted scores of SO posts and the posts' compliance with coding style rules are correlated.

Table III
MEAN AND MEDIAN PEARSON CORRELATION VALUES FOR POSTS IN DIFFERENT RANGES OF POST SCORES

Range	Mean		Median	
	r	p	r	p
[-10, +20]	-0.870	8.591e-10	-0.821	5.025e-08
[-10, +25]	-0.755	2.503e-07	-0.732	8.634e-07
[-10, +30]	-0.731	1.256e-07	-0.651	7.127e-06
[-10, +35]	-0.748	1.221e-08	-0.630	7.969e-06
[-10, +40]	-0.600	2.129e-05	-0.631	5.83e-06

D. RQ4: Compliance and Reputation

To investigate if SO users' reputation correlates with their coding style compliance, we filtered out collaboratively written code snippets by only considering the posts last edited by their original author. Additionally, we only considered users who have written more than 5 posts to get more reasonable average values. We obtained the average number of violations per statement by grouping the snippets by their author and computing the average over them.

In contrast to RQ3, we found that there seems to be no correlation between coding style compliance and user reputation (r = -0.0532680, p = 0.007529). As Figure 3 illustrates, the values are seemingly arbitrarily distributed. Interestingly, we discovered a noticeable sparseness of data points for users with a reputation around 500. This phenomenon, however, stems from the original dataset and is not related to our research focus.

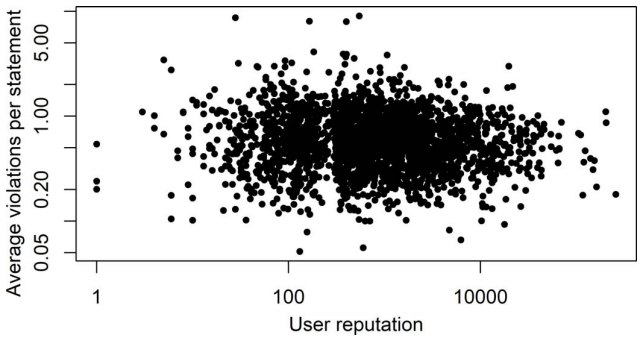

Figure 3. Scatter plot of average violations per statement over user reputation

VI. Discussion

The goal of this study was to analyse the patterns and effects of coding style compliance on SO. We found that Python code on SO generally does not comply with coding style guides, but there is a striking correlation between coding style compliance and post popularity: On average, posts with fewer violations per statement are favoured by the community. However, somewhat counterintuitively, there is no correlation between coding style compliance and user reputation.

The low coding style compliance did not come as a complete surprise to us. Users generally intend to convey solutions quickly and straightforwardly. Additionally, the SO editor lacks the formatting support offered by advanced IDEs. Moreover, snippets on SO are usually not complete Python scripts, just fragments, often containing placeholders for code or identifiers.

In general, one could challenge the applicability of coding style guides in the context of SO. Good coding style might help to deliver messages effectively and decrease cognitive load, probably a cause of highly voted posts having better coding style compliances. However, many style rules might not be reasonable, and the relevance of compliance debatable. Snippets are often made concise by leaving out unnecessary details [9] which causes compliance violations – conciseness may be more important than compliance.

Because Python snippets on SO do usually not adhere to coding style conventions, they may not be considered representative of good coding style. Therefore, applying mining or learning approaches [30], [17] to code snippets on SO would not necessarily produce the desired results.

VII. Threats to Validity

A. Internal Validity

Identification of Python Code Snippets: We only processed posts containing the substring 'python' in any of their tags. However, some posts are not correctly tagged or contain mixed programming languages, so we may have included snippets that are not actually Python code and excluded snippets that are. Therefore we processed all supposed Python code snippets with Pylint, discarding the ones that could not be processed. While we manually verified many results, there may be non-Python code that it incorrectly accepts.

Extreme Values, Noise and Distortion: The dataset is dominated by extremes. The outliers tend to distort the general result, which is why filtering was needed in order to obtain meaningful results. Most importantly, we discarded snippets that are shorter in length than 6 lines or 6 statements (a size commonly used [28]), as we found a lot of the problematic results that distort the general values in very short snippets. However, this again introduces a threat to validity, as a great amount of the original data gets filtered out along the way.

Violations in Incomplete Snippets: The majority of snippets on SO are incomplete Python scripts. While this might be enough to convey the key ideas to human readers, it causes many coding convention violations, e.g., usage of undeclared variables. Such rules, when discovered, were disabled in Pylint. However, the Pylint configuration was manually decided based on our best knowledge. Potentially, other researchers would have chosen different violations to ignore.

Allocation of Code Authorship: After a post owner originally created a post, any user can edit it. Therefore, as we use only the most recent version of the code snippets, it is possible that other users have modified the code in the meantime. We only consider posts where the last editor is also the original author to mitigate the problem of wrongly allocated authorship. It is possible that changes were made by other users in-between the creation and the last edit that changed the code.

B. External Validity

Our study is limited to Python snippets on SO and may not be generalised to other languages. Moreover, snippets found on other platforms similar to SO may exhibit a different pattern of coding style compliance.

VIII. Conclusion and Future Work

We fetched over 1.9 million Python-tagged code snippets with their relevant metadata. From them, we extracted 407,097 Python snippets of a least 6 lines and at least 6 statements for which we obtained coding style compliance data via Pylint. Our data set is available online [29].

Our results show that 93.87% of the code snippets contain coding style violations, with an average of 0.7 violations per statement. We also found that the 10 most common style violations make up 85.1% of the total, where *bad-whitespace* dominates with 33.4% of all violations.

Furthermore, we found that compliance with coding style guides does affect users' perceptions of posts, as we discovered a strong correlation between the vote score a post receives and the average number of coding style violations per statement of its contained snippets. For posts with vote scores between -10 and +20, the correlation coefficient r is -0.87 ($p < 10^{-7}$), implying high significance. Finally, we saw that the author's reputation is not an indicator of a post's compliance with coding conventions since they do not exhibit correlation.

Our findings suggest that Python code snippets on Stack Overflow do not represent good coding style and may not be suitable for mining or learning tasks. Future work should investigate whether the same can be observed for other languages.

References

[1] S. Baltes, L. Dumani, C. Treude, and S. Diehl, "SOTorrent: Reconstructing and analyzing the evolution of Stack Overflow posts," in *Proceedings of the 15th International Conference on Mining Software Repositories*, 2018, pp. 319–330.

[2] F. Fischer, K. Böttinger, H. Xiao, C. Stransky, Y. Acar, M. Backes, and S. Fahl, "Stack Overflow considered harmful? the impact of copy&paste on Android application security," in *IEEE Symposium on Security and Privacy (SP)*. IEEE, 2017, pp. 121–136.

[3] L. An, O. Mlouki, F. Khomh, and G. Antoniol, "Stack Overflow: a code laundering platform?" in *24th International Conference on Software Analysis, Evolution and Reengineering (SANER)*. IEEE, 2017, pp. 283–293.

[4] S. Baltes and S. Diehl, "Usage and attribution of Stack Overflow code snippets in GitHub projects," *Empirical Software Engineering*, Oct 2018.

[5] R. C. Martin, *Clean code: a handbook of agile software craftsmanship*. Upper Saddle River, NJ: Prentice Hall, 2008.

[6] D. Spinellis, "Elyts edoc," *IEEE Software*, vol. 28, no. 2, pp. 104–104, 2011.

[7] PEP 20 – The Zen of Python. [Online]. Available: https://www.python.org/dev/peps/pep-0020/

[8] K. Hart and A. Sarma, "Perceptions of answer quality in an online technical question and answer forum," in *Proceedings of the 7th International Workshop on Cooperative and Human Aspects of Software Engineering (CHASE)*, 2014, pp. 103–106.

[9] S. M. Nasehi, J. Sillito, F. Maurer, and C. Burns, "What makes a good code example? A study of programming Q&A in StackOverflow," in *28th IEEE International Conference on Software Maintenance (ICSM)*, 2012, pp. 25–34.

[10] M. Duijn, A. Kučera, and A. Bacchelli, "Quality questions need quality code: classifying code fragments on stack overflow," in *Proceedings of the 12th Working Conference on Mining Software Repositories*. IEEE Press, 2015, pp. 410–413.

[11] C. Treude and M. P. Robillard, "Understanding stack overflow code fragments," in *International Conference on Software Maintenance and Evolution (ICSME)*. IEEE, 2017, pp. 509–513.

[12] S. Baltes, C. Treude, and S. Diehl, "SOTorrent: Studying the origin, evolution, and usage of Stack Overflow code snippets," in *Proceedings of the 16th International Conference on Mining Software Repositories (MSR)*, 2019.

[13] T. Lee, J. B. Lee, and H. P. In, "A study of different coding styles affecting code readability," *International Journal of Software Engineering and Its Applications*, vol. 7, no. 5, pp. 413–422, 2013.

[14] N. C. Zakas. (2012) Why coding style matters. [Online]. Available: https://www.smashingmagazine.com/2012/10/why-coding-style-matters/

[15] M. O. Elish and J. Offutt, "The adherence of open source Java programmers to standard coding practices," in *6th IASTED International Conference on Software Engineering and Applications (SEA)*, 2002, pp. pp. 193–198.

[16] M. Smit, B. Gergel, H. J. Hoover, and E. Stroulia, "Code convention adherence in evolving software," in *International Conference on Software Maintenance (ICSM)*, 2011.

[17] M. Allamanis, E. T. Barr, C. Bird, and C. Sutton, "Learning natural coding conventions," in *Proceedings of the 22nd ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE)*, 2014, pp. 281–293.

[18] S. Dasgupta and S. Hooshangi, "Code quality: Examining the efficacy of automated tools," in *23rd Americas Conference on Information Systems (AMCIS)*. AIS, 2017.

[19] L. Gong, M. Pradel, M. Sridharan, and K. Sen, "DLint: dynamically checking bad coding practices in JavaScript," in *Proceedings of the 2015 International Symposium on Software Testing and Analysis (ISSTA)*, 2015, pp. 94–105.

[20] K. Daimi, S. Banitaan, and K. Liszka, "Examining the performance of Java static analyzers," in *Proceedings of the International Conference on Software Engineering Research and Practice (SERP)*, 2013.

[21] Airbnb JavaScript style guide. [Online]. Available: http://airbnb.io/javascript/

[22] Google C++ style guide. [Online]. Available: https://google.github.io/styleguide/cppguide.html

[23] PEP 8: The style guide for Python code. [Online]. Available: https://pep8.org/

[24] B. Morelli. (2017) 5 JavaScript style guides – including AirBnB, GitHub, & Google. [Online]. Available: https://codeburst.io/5-javascript-style-guides-including-airbnb-github-google-88cbc6b2b7aa

[25] Google Python style guide. Original-date: 2015-05-20T19:18:59Z. [Online]. Available: http://google.github.io/styleguide/pyguide.html

[26] Stack Overflow trends – most popular languages. [Online]. Available: https://insights.stackoverflow.com/trends?tags=java%2Cc%2Cc%2B%2B%2Cpython%2Cc%23%2Cvb.net%2Cjavascript%2Cassembly%2Cphp%2Cperl%2Cruby%2Cswift%2Cr%2Cobjective-c

[27] D. Yang, A. Hussain, and C. V. Lopes, "From query to usable code: an analysis of stack overflow code snippets," in *Proceedings of the 13th International Conference on Mining Software Repositories*. ACM, 2016, pp. 391–402.

[28] M. Gabel and Z. Su, "A study of the uniqueness of source code," in *Proceedings of the 18th ACM SIGSOFT International Symposium on Foundations of Software Engineering (FSE)*, 2010, pp. 147–156.

[29] Pylint results for Python code snippets on Stack Overflow. [Online]. Available: https://zenodo.org/record/2558544

[30] S. Butler, M. Wermelinger, Y. Yu, and H. Sharp, "Mining Java class naming conventions," in *27th IEEE International Conference on Software Maintenance (ICSM)*. IEEE, 2011, pp. 93–102.

[31] Pylint 1.9.3 documentation – pylint user manual. [Online]. Available: http://pylint.pycqa.org/en/1.9/