An Analysis of Python’s Topics, Trends, and Technologies Through Mining Stack Overflow Discussions

Hamed Tahmooresi, Abbas Heydarnoori, and Alireza Aghamohammadi

Abstract—Python is a popular, widely used, and general-purpose programming language. In spite of its ever-growing community, researchers have not performed much analysis on Python’s topics, trends, and technologies which provides insights for developers about Python community trends and main issues. In this article, we examine the main topics related to this language being discussed by developers on one of the most popular Q&A websites, Stack Overflow, as well as temporal trends through mining 2 461 876 posts. To be more useful for the software engineers, we study what Python provides as the alternative to popular technologies offered by common programming languages like Java. Our results indicate that discussions about Python standard features, web programming, and scientific programming are the most popular areas in the Python community. At the same time, areas related to the scientific programming are steadily receiving more attention from the Python developers.

Index Terms—Python, Q&A websites, Stack Overflow, Topic Modeling, Trend Analysis.

1 INTRODUCTION

Python is a widely used, high-level, general-purpose, interpreted, and dynamic programming language [1]. According to Stack Overflow annual survey engaging more than 90 000 developers in 2019, Python just edged out Java in overall ranking at the end of 2018, much like it edged out C# in 2017, and PHP in 2016 alike. Stack Overflow called Python the fast-growing major programming language [2]. In spite of the prevalence of Python, researchers have not adequately focused on analyzing the trends and technologies of this language in software developer communities, and Question and Answer (Q&A) websites such as Stack Overflow which is the de facto website actively used by developers. Analyzing this invaluable resource of information can help developers to gain insight about Python’s trends and main issues. Prior studies have leveraged a topic modeling approach called Latent Dirichlet Allocation (LDA) [3] to categorize discussions in specific areas such as web programming [4], mobile application development [5], security [6], and blockchain [7]. However, none of them has focused on Python. In this study, we shed light on the Python’s main areas of discussion through mining 2 461 876 posts, from August 2008 to January 2019, using LDA based topic modeling. We also analyze the temporal trends of the extracted topics to find out how the developers’ interest is changing over the time. After explaining the topics and trends of Python, we investigate the technologies offered by Python, which are not properly revealed by the topic modeling. Because of the significant prevalence of Python, software engineers, mostly the experts of other programming languages, may be eager to know the technologies of their interest provided by Python as the alternative to investigate this language further. That expert may search online resources and read a lot to find an appropriate answer. However, the aggregation and verification of the obtained knowledge may be on her own. As a result, by leveraging the word embedding model [8], we can find out for each technology provided for other programming languages what technologies are offered by Python which are practically employed by Python developers. Hence, we pave the way for better understanding Python’s solutions. The main findings from our study indicate that Python developers mostly discuss Python standard features, web programming, and scientific programming. However, the popularity of scientific programming is growing faster.

2 STUDY SETUP

The primary goal of our study is to extract the areas that Python developers discuss on Stack Overflow. For this purpose, we introduce three research questions:

RQ1: What are the discussion areas in the Python community?
RQ2: How is the interest of Python developers changing over the time?
RQ3: What technologies does Python provide?

Figure 1 illustrates the high-level steps we take to answer our research questions.
2.1 RQ1: What are the discussion areas in the Python community?

To address RQ1, we first extracted the tags associated with each language (Step 1). Similar to prior studies, we identified discussions of each language according to Stack Overflow’s tag mechanism [9], [10]. In this website, the owner of a question may assign, up to five tags as keywords to each question which denotes the technological categories of that question [10]. We preprocessed Python posts extracted (Step 2) as the input of our topic modeling.

We pruned questions with the minus or zero score without any accepted answers in order to improve the overall quality of the topic modeling (Step 3). Due to the scoring mechanism, these questions have poor quality [4]. This way, 9% of all the Python posts were pruned from our dataset. Next, we cleansed a textual content of the extracted posts (Step 4) as follows:

1) All code snippets enclosed in the <code> tag were discarded since the source code would introduce noise in the results of the topic modeling [11].

2) All HTML tags were removed as well (e.g., <a href="...">, <b>, and so forth).

3) We tokenized the corpus and removed common English-language stop words such as a, the, and is which do not help creating meaningful topics.

4) All the tokens were stemmed using the Porter stemming algorithm [12].

Afterwards, we exploited the popular topic modeling technique, LDA, to investigate the developers’ discussion areas (Step 5). LDA is an unsupervised model for performing statistical topic modeling that uses a bag of words approach [3]. Recently, LDA has been employed in software engineering communities such as Stack Overflow to extract topics discussed in the crowd [4], [5]. It takes the number of topics $K$ as an input. Larger values of $K$ will result in fine-grained topics and lower values will produce coarse-grained topics. Unfortunately, by choosing a small $K$ (e.g., 10 or 20), many discussion areas may remain hidden. As an example, an LDA model with 100 topics for Python discussions reveals topics such as game programming, IoT, and testing. However, another model with 20 topics suppresses these areas. On the other hand, as the number of topics increases, their visualization and analysis become hard to understand. Besides, many duplicates emerge among topics, and thus a manual merge is required [5]. Therefore, we first generated a LDA model with 100 topics and then asked two Python experts — who are not the authors of this paper — with more than seven years of experience to manually merge the LDA model.

The experts observed that topics extracted from Stack Overflow usually obtain known technologies among their top probable words (e.g., XML, server, web framework names, and so forth). Therefore, if two topics share similar words having high occurrence probability, which match the name of known technologies, they may be good candidates for being merged. For example, if two topics have the word django — a Python web framework — as a highly probable word, we can anticipate that they are both about web programming subject. As a result, the experts considered Stack Overflow tags as a source of words resembling technologies to help merge topics more accurately. Having examined the Stack Overflow tags along with fine-grained topics, our experts decided to merge topics into 12 clusters (Step 6). Note that, the experts operated independent of each other. Then, they shared the results. In the case of difference, they discussed until all of the disagreements were resolved.

2.2 RQ2: How is the interest of Python developers changing over the time?

To address RQ2, we partitioned the dataset (obtained in Step 2) into three-month time intervals (Step 7). Three-month intervals yielded enough discussions in each interval which make our analysis more reliable. Since the Python community is constantly growing, we observed that over the time, the frequency of each cluster is increasing as well. Thus, to better analyze the data, we borrowed the concept...
of impact of a topic presented by Barua et al. [10] to obtain the portion of a topic $t_k$ in a time interval $\text{intv}$:

$$\text{impact}(t_k, \text{intv}) = \frac{1}{D(\text{intv})} \sum_{d_j \in D(\text{intv})} \theta(d_j, t_k) \quad \text{(Eq. 1)}$$

Where $D(\text{intv})$ is the set of all posts in the time interval $\text{intv}$. $\theta(d_j, t_k)$ denotes the probability of a particular topic $t_k$ occurring in the document $d_j$. In order for calculating the impact of each cluster in an interval, we simply sum up the impact of its topic.

Note that, a decrease in the impact of a topic does not mean a decline in discussions related to it. In fact, $\text{impact}(t_k, \text{intv})$ resembles the portion of a topic in an interval to other topics as Equation (1) contains a fraction over $D(\text{intv})$. Therefore, if other topics grow faster, the topic loses its portion. Now we can analyze the interest of developers on each topic cluster over the time (Step 8).

We exploited the MK test to find an increase or decrease in trends of the clusters. The MK test is a non-parametric statistical method which assesses the existence of a monotonic increasing or decreasing trend (either linear or nonlinear) for a variable over the time [13].

2.3 RQ3: What technologies does Python provide?

We consider technology as software solutions, packages, libraries, and frameworks provided for developers in a programming language such as Django [5]. In this section, we focus on Python’s technologies as alternatives to popular solutions provided by common programming language like Java. Hence, experts of the other programming languages can better get familiar with Python technologies which are considered alternatives to the technologies they use. To this aim, we exploit the same idea suggested by Chen et al. [14] that uses the crowd knowledge to recommend corresponding technologies in different programming languages. The proposed approach is based on word2vec, a computationally efficient predictive model for learning word embedding from a raw text. What word2vec produces is a vector for each word in the corpus [8]. One of the interesting features of the model is to organize technologies and comprehend the implicit relationships between them via unsupervised learning [8]. For instance, suppose $V_{\text{php}}, V_{\text{py}}, V_{\text{py}},$ and $V_d$ be the vector representations of the word PHP, Laravel, Python, and Django respectively. The results of the calculation $V_i - V_{\text{php}} + V_{\text{py}}$ is closer to $V_d$ than any other word vectors. That is to say, not only do Django and Laravel have clusters near each other, but they each have similar distances in a vector space to the programming language whose web frameworks they are. Therefore, without even specifying any context (being the web framework of a programming language in this example), the algorithm can extract the latent relationships between languages. We use the Stack Overflow’s tag mechanism as developers use it to elaborate upon their libraries, frameworks, and technical concepts to which their questions are directly relative, in order to be easily found by respondents [15]. Leveraging the tag mechanism to describe technologies in the software development community has been recently performed by Barua et al. [10]. Thus, we created a new dataset (Step 9) by using tags of each question. For example, if a question is assigned the tags $<\text{java, swing, jframe}>$, we simply consider “java swing jframe” as an item of our dataset. Finally, we trained our word embedding model using our dataset (Step 10).

3 Results

Herein, we present the results and findings of our study.

3.1 Areas Discussed by Python Developers

As mentioned before, we used LDA to extract fine-grained areas discussed. Next, two experts grouped the topics into 12 clusters. Extracted Python clusters along with their portions are demonstrated in Figure 2. The most frequent area belongs to standard features and problems related to the Python language itself. This may be due to the fact that the Python community contains a significant number of amateur developers who have started Python without enough programming knowledge. The second and third most frequent discussion areas belong to the web programming and scientific programming. Furthermore, OS, multithreading, message queuing issues, data format (e.g., JSON, XML, CSV, etc.) and serialization are also widespread among developers. Interestingly, Python developers are attracted to game programming and issues about programming on devices such as Rasberri Pi and Arduino which are popular in the IoT community.

Since we grouped topics into clusters, we can now simply drill a step down to the finer-grained areas. For example, Figure 3 illustrates topics of the Python standard features (The first bar in Figure 2). According to Figure 3, data structures, working with strings, list comprehension and generators, package management, and installation are the most
3.2 Temporal Trends

As for temporal trend analysis, we plotted only clusters having a significant trend (increasing or decreasing, according to MK test) to obtain more beneficial results. As shown in Figure 4, discussions about Python standard features and web programming are losing share among other areas. As a possible explanation, these two areas have been discussed from the very beginning of Stack Overflow, at the time when other areas were obscure. However, as other areas arise, the portions of the questions related to web programming and Python standard features decline over the time. Besides, questions related to these two areas become saturated and most of the challenges are discussed which may amplify this decline.

Study Findings. Scientific programming is increasing rapidly, exceeding web programming. Challenges related to packaging, library versioning, and installation have remained stable since 2015. Another finding is that subject related to OS, multitasking, and message queuing is rising as well as data formats and serialization, especially from the start of 2015.

3.3 Python Technologies

Leveraging Chen et al.’s work [14], we can now extract Python’s technologies and organize them according to their correspondence to solutions offered by other programming languages. Table 1 shows a sample of the Python technologies recommended by the approach. However, developers can use our trained model (which is based on the approach offered by Chen et al.’s work [14]) to further investigate Python’s technologies.² As a simple example, given Visual Studio — the C# IDE — as an input, the approach suggests Pycharm a popular Python IDE as a solution to the needs Visual Studio satisfies in the C# community. As a more advanced example, the model recommends “<Virtualenv, pip, requirements.txt>” as an alternative to Maven (in Java). In reality, Python developers list their library dependencies in a file so-called requirements.txt (analogous to pom.xml in Maven). Furthermore, by using pip, all dependencies can be downloaded from the pip online repositories (analogous to Maven repositories) to a directory located in the host machine. However, there come two main differences between the Python’s and Java’s way of building applications:

1) Unlike Maven, pip does not keep multiple versions of the same package.
2) All applications in the same environment consider a single directory as a source code of external packages. While in Java, each application has its lib directory in which maintains a copy of libraries that depends on Jar files.

As a result, there is no way to have multiple applications depending on different versions of the same package in a single machine. Here comes a popular tool, Virtualenv, to create multiple virtual environments on the same machine. Each has its own Python interpreter, site-package, pip tool, and so forth.

Note that, experts of the other programming languages may use the Google search engine to find their correspondences offered by Python. For example, if we search “Java Maven alternative in Python” the first recommended page would be a Stack Overflow question that its accepted answer only mentions distutils and setuptools. In fact, the answer only covers packaging tools in Python and skips dependency management. Furthermore, Pybuilder is recommended in topic links, which is not popular as at the time of writing this paper. The frequency of its corresponding tag in the Stack Overflow is just 43. However, using the word embedding approach will extract technologies which are real alternatives and practically used according to millions of discussions in the crowd [14].

². https://goo.gl/8Ne2cK
### Table 1
The catalog of Python’s solutions including their correspondence to technologies in other languages.

| Technology | Context | Python’s alternatives |
|------------|---------|-----------------------|
| **Java**   |         |                       |
| Hibernate  | ORM     | SQLAlchemy, Django-models |
| Maven      | Build tool | Virtualenv, pip, requirements.txt |
| Swing      | Desktop application development | PyQt, WXPython, Tkinter |
| Stanford-NLP | Natural language processing | NLTK, Gensim |
| Jackson    | JSON library & object serializer | Pickle, SimpleJson, Django-serializer |
| Spring     | Dependency injection, Module integration, Web development | Django, uwsgi, tornado, celery |
| Spring-MVC | Web programming | Django, Flask |
| JAR        | Packaging | PyInstaller, Py2exe, Egg |
| Jackson    | JSON library and object serializer | Pickle, simpleJson, Django-serializer |
| Eclipse    | Integrated development environment (IDE) | Pycharm |
| Tomcat     | Web server | Uwsgi, gunicorn, tornado, Nginx |
| **PHP**    |         |                       |
| Smarty     | Template engine | Django-templates, Jinja2, Cheetah |
| PHPMailer  | Mail sending | Sendmail, SMPTLib |
| Laravel    | Web programming | Django, Flask |
| WordPress  | Content management system | Django-CMS, Mezzanine |
| PDO        | Data-access abstraction layer | MySQL-python, pymssql, psycopg2 |
| cURL       | transferring data using various protocols such as HTTP, FTP, and so forth. | Python-requests |
| **C#**     |         |                       |
| NUnit      | Unit testing framework | Nose, Python-unittest |
| Visual Studio | Integrated Development Environment | Pycharm |
| Entity- framework | Object Relational Mapping (ORM) | Django-queryset, sqlachemy |
| ASP.NET-MVC | Web application development | Django, Flask |
| IIS        | Web server | Uwsgi, gunicorn, tonado, Nginx |
| WFP        | Rendering user interfaces in Windows-based applications | PyQt, WXPython, Tkinter |
| Unity3d    | Game engine | Pygame |
| **C & C++**|         |                       |
| Ruby       | Package management | pip, Anaconda, virtualenv |
| ActiveRecord | Object-relational mapping (ORM) | Django-queryset, sqlachemy |
| Cucumber   | Test & Behavior driven development | Lettuce, Robotframework |
| Devise     | Authentication framework | Django-allauth, Django-authentication, Django-registration |
| **Javascript** |         |                       |
| NPM        | Package management | Pip, Anaconda, virtualenv |
| Socket.io  | Realtime web application development | Tornado, gevent |
| Mongoose   | MongoDB ODM (Object Document Mapper) | Pymongo |
| Sequelize  | Object Relational Mapping (ORM) | SQLAlchemy, Django-models |

### 4 Related Work

Several studies have been performed using Stack Overflow for a wide variety of purposes. Researchers have employed LDA to categorize discussions in specific areas such as web programming [4], mobile application development [5], security [6], and blockchain [7]. For example, Barua et al. used LDA to extract main topics discussed on Stack Overflow [10]. They also investigated how developer’s interest would change over the time using the impact of a topic. Finally, to provide a more focused view, they extracted the change of interest in specific technologies over the time.

Rosen et al. concentrated on mobile developers on Stack Overflow which exploited LDA to extract main topics and most difficult issues of mobile developers [5]. However, none of the prior studies focused on Python despite of its ever-growing popularity.

### 5 Conclusion

In this article, we investigated topics and technologies of the Python programming language. We employed trend analysis and used a clustering approach for improving the understandability of a large number of topics. Furthermore,
we used an approach based on the word2vec model to recommend Python’s solutions corresponding to technologies of other programming languages.

Our results indicate that standard features provided by Python, web development, and scientific programming are the most popular areas among Python developers on Stack Overflow. However, scientific programming is gaining more popularity. Ultimately, using the word2vec model we extracted some of the Python’s technologies as alternatives to technologies offered by other programming languages.

REFERENCES

[1] J. M. Redondo and F. Ortin, “A comprehensive evaluation of common python implementations,” IEEE Software, vol. 32, no. 4, pp. 76–84, 2015.
[2] “Developer Survey Results,” https://insights.stackoverflow.com/survey/2019, online; accessed 3 July 2019.
[3] D. M. Blei, A. Y. Ng, and M. I. Jordan, “Latent dirichlet allocation,” Journal of machine Learning research., vol. 3, no. Jan, pp. 993–1022, 2003.
[4] K. Bajaj, K. Pattabiraman, and A. Mesbah, “Mining questions asked by web developers,” in Proceedings of the 11th Working Conference on Mining Software Repositories, MSR. ACM, 2014, pp. 112–121.
[5] C. Rosen and E. Shihab, “What are mobile developers asking about? A large scale study using stack overflow,” Empirical Software Engineering., vol. 21, no. 3, pp. 1192–1223, 2016.
[6] X. Yang, D. Lo, X. Xia, Z. Wan, and J. Sun, “What security questions do developers ask? A large-scale study of stack overflow posts,” J. Comput. Sci. Technol., vol. 31, no. 5, pp. 910–924, 2016.
[7] Z. Wan, X. Xia, and A. E. Hassan, “What is discussed about blockchain? a case study on the use of balanced lda and the reference architecture of a domain to capture online discussions about blockchain platforms across the stack exchange communities,” IEEE Transactions on Software Engineering, 2019.
[8] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, “Distributed representations of words and phrases and their compositionality,” in Proceedings of the 27th Annual Conference on Neural Information Processing Systems, 2013, pp. 3111–3119.
[9] M. Allamanis and C. A. Sutton, “Why, when, and what: analyzing stack overflow questions by topic, type, and code,” in Proceedings of the 10th Annual Conference on Mining Software Repositories, 2013, pp. 53–56.
[10] A. Barua, S. W. Thomas, and A. E. Hassan, “What are developers talking about? an analysis of topics and trends in stack overflow,” Empirical Software Engineering., vol. 19, no. 3, pp. 619–654, 2014.
[11] S. W. Thomas, “Mining software repositories using topic models,” in Proceedings of the 33rd International Conference on Software Engineering, ICSE. ACM, 2011, pp. 1138–1139.
[12] M. F. Porter, “An algorithm for suffix stripping,” Program, vol. 40, no. 3, pp. 211–218, 2006.
[13] H. B. Mann, “Nonparametric tests against trend,” Econometrica: journal of the Econometric Society, pp. 245–259, 1945.
[14] C. Chen, Z. Xing, and Y. Liu, “What’s spain’s paris? mining analogical libraries from q&a discussions,” Empirical Software Engineering, vol. 24, no. 3, pp. 1155–1194, 2019.
[15] C. Chen and Z. Xing, “Mining technology landscape from stack overflow,” in Proceedings of the 10th International Symposium on Empirical Software Engineering and Measurement, 2016, pp. 14:1–14:10.

Hamed Tahmooresi is a PhD student at the Sharif University of Technology. His research interests include software engineering, software architecture and design, and mining software repositories. Contact him at tahmooresi@ce.sharif.edu.

Abbas Heydarnoori is an assistant professor in the Department of Computer Engineering at the Sharif University of Technology. Before, he was a post-doctoral fellow at the University of Lugano, Switzerland, working with Prof. Walter Binder. Abbas did his PhD in the School of Computer Science at the University of Waterloo, Canada, under the supervision of Prof. Krzysztof Czarnecki. His research interests focus on software evolution and maintenance, mining software repositories, and recommendation systems in software engineering. Contact him at heydarnoori@sharif.edu.

Alireza Aghamohammadi is a Ph.D. student at Sharif University of Technology (SUT). He works on a wide range of recommendation systems that exploit state-of-the-art machine learning techniques. He is also a software developer for more than seven years. Contact him at aaghamohammadi@ce.sharif.edu.