What Topics Do Developers Concern? An Analysis of Java Related Posts on Stack Overflow

Peng Zhang
School of Software Technology
Dalian University of Technology
Dalian, China
e-mail: zp0016@qq.com

***Abstract*—As a popular on-line question and answer site [1], Stack Overflow has attracted millions of developers to communicate, collaborate, and share knowledge. Aiming to better understand the trend of Java, one of the most popular programming languages in the world, this study investigates millions of Java-related posts on Stack Overflow. In this study, we first employ Latent Dirichlet Allocation (LDA) to detect topics among Java-related posts. Then, we conduct further analysis over these detected topics, e.g., the most popular topic and the most difficult topic to answer. As a result, we detect 40 topics and top 10 key terms. The top four popular topics include "System Platform", "Java Archive", "Import Sources", and "Maven Build". Meanwhile, "List", "Web Development", "JavaFX", and "User Security" are the top four difficult topics. Our study provides new insights into Java-related topics on Stack Overflow.**

Keywords-stack overflow; Java; topic model; latent dirichlet allocation

I. INTRODUCTION

Stack Overflow is a well-known platform of technical question and answer. With Stack Overflow, developers provide answers to some technical questions by their own knowledge and experience [2 and 3]. In the 2 years since its foundation in 2008, more than 1 million questions have been asked on Stack Overflow, and more than 2.5 million answers have been provided [4]. To date, Stack Overflow turns into the repository of the knowledge about software engineering [5 and 6]. Hence, we can achieve a lot of valuable information from such a knowledge repository, such as the use of specific technology or the trend discussed by developers [2]. By mining Stack Overflow, a lot of interesting and useful conclusions could be achieved. Stack Overflow has been studied and analyzed by some researchers [7]. Many previous works focus on some specific posts [2], including the posts related to security technology, the posts related to mobile development, etc. [2, 8, and 9].

In this study, we choose the posts related to Java to analyze. We first use some heuristics to extract posts related to Java from Stack Overflow and preprocess the texts of all extracted posts. Then, we employ the LDA (Latent Dirichlet Allocation) topic model to find the topics discussed by developers automatically [10]. Finally, we analyze the topics of these findings and explore the relationship between topics and posts. At the same time, we also analyze the topics related to the questions which are very popular and the topics which are difficult to answer.

Our empirical study investigates the following three Research Questions (RQs) and achieve the answers to these RQs.

RQ1. What topics related to Java are asked on Stack Overflow?

We use a topic model to investigate the topics related to Java and explore the relationship between topics and posts. Questions related to Java on Stack Overflow cover a wide range of topics.

RQ2. Which topics are the most popular among the questions related to Java?

We measure the popularity of topics related to Java by one major metric (i.e., the average number of views), and three minor metrics (i.e., the average number of comments, the average number of favorites, and the average score). The top four most popular topics are "System Platform", "Java Archive", "Import Sources" and "Maven Build".

RQ3. Which topics related to Java are the most difficult to answer?

We measure the difficulty of topics related to Java by two metrics (i.e., the average time needed for an accepted answer and the average proportion of the number of answers to the number of views). The top four most difficult topics are "List", "Web Development", "JavaFX", and "User Security".

The remainder of this paper is structured as follows. We introduce the Stack Overflow posts and Latent Dirichlet Allocation (LDA) in Section 2. The data collection step and data preprocessing step are described in Section 3 respectively. Section 4 introduces the use of topic model tools. Our experimental results are presented and discussed in Sections 5. Section 6 concludes the paper.

II. PRELIMINARIES

In this section, we introduce the background knowledge of Stack Overflow and posts on it. Then, we present a brief introduction to Latent Dirichlet Allocation (LDA), a classic topic model.

A. How Questions are Asked on Stack Overflow

Stack Overflow is similar to most Q&A sites by allowing users to submit posts (questions and answers), comment on posts, and vote on posts [8]. When a user submits posts, they can tag them with their subject area so that others can detect

978-1-7281-0831-5/19/$31.00 ©2019 IEEE

posts more easily. Each question must have at least one label and up to five different labels to identify its subject area.

B. Posts on Stack Overflow

Stack Overflow consists of millions of posts covering a wide range of topics, such as programming related, mobile-related, and security-related topics. For instance, Fig. 1 presents a Java-related post which was asked on Stack Overflow, the reporter submits the tile as "How to declare an array of method references?" and gives detailed description about this question in the body of the post. Participators can provide answers and suggestions under the body and make a full communication by posting comments. In addition, some other important information is included in Fig. 1, including the tags of the post which represent the subject areas the post belonging to, the number of comments, and the edit date etc. In such a way, over millions of these posts formed a larger knowledge warehouse in Stack Overflow, which provide effective support for our development.

C. Topic Model

A topic model is a probability distribution model which models a document as a topic vector. In the vector, each dimension means the probability that the document belongs to this topic. In the literature, many topic modeling techniques exist for different data types, assumptions, and goals. In this paper, we use a well-known topic modeling technique, namely Latent Dirichlet Allocation (LDA). In the LDA model, each document consists of a mixture of topics. Topics are allowed to exist across multiple documents, thus making it possible for the LDA model to discover themes and ideas among documents [9].

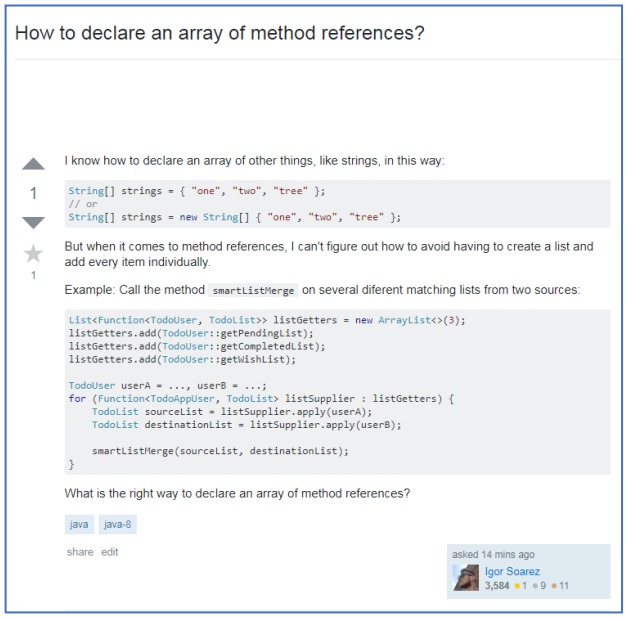

Figure 1. A question (post) asked on stack overflow.

III. Data Extraction and Processing

To analyze the posts in Stack Overflow, in this section we first explain the process of extracting the Stack Overflow data (subsection 3.1), and then detail the way to process these data (subsection 3.2).

A. Data Extraction

To understand the discussion topics from the Stack Overflow posts, in this study, we need to collect data involved in posts, which are publicly available in XML format provided by Stack Exchange. There are eight XML documents in Stack Exchange, including Badges.xml, Comments.xml, PostHistory.xml, PostLinks.xml, Posts.xml, Tags.xml, Users.xml and Votes.xml. We mainly use Posts.xml in this study for that it contains the main context of posts and other metadata liking view count, post type (user generated question posts and answer posts), creation date, and ID of the user etc. More detailed post information is shown in Table 1 [2]. First, we traverse posts from 2008 to 2017. In this process, 34,857,917 posts are collected, including 13,472,796 (38.6%) question posts and 21,299,522 (61.1%) answer posts. There are also 85,599 (0.03%) other type posts which we do not pay attention to. Second, we need to identify Java-related posts. This step involves three sub-steps as following:

TABLE I. Detailed Information of a Post

Name	Quantity
Id	ID of the post
PostTypeId	Type of the post: 1 represents a question post, and 2 represents an answer post
AcceptedAnswerId	ID of the corresponding accepted answer post for the question post (optional, and appears only when PostTypeId==1)
ParentId	ID of the corresponding question post for the answer post (optional, and appears only when PostTypeId==2)
CreationDate	Creation date of the post
Score	Average score by the viewers for the post
ViewCount	Total number of views for the post (optional, and appears only when PostTypeId==1)
Body	Body of the post
OwnerUserId	ID of the post owner (optional)
LastEditorUserId	ID of the person who last edited the post
LastEditorDisplayName	Username of the person who last edited the post
LastEditDate	Date when the post is last edited
LastActivityDate	Date when the status of the post is last changed
Title	Title of the post (optional, and appears only when PostTypeId==1)
Tags	Tags of the post (optional, and appears only when PostTypeId==1)
AnswerCount	Number of answers for the post (optional, and appears only when PostTypeId==1)
CommentCount	Number of comments for the post
FavoriteCount	Number of people who like the post (optional, and appears only when PostTypeId==1)
CommunityOwnedDate	Date when the post is owned by community (optional)
OwnerDisplayName	Username of the post owner (optional)
ClosedDate	Date when the post is closed (optional)

1) Obtaining candidate tags related to Java

Each question post and related answer post have same tags, we collect all these posts' tags. Then, we select tags whose body contain the keyword of "Java", and we refer to these tags as candidate tags.

As shown in Fig. 2, we take the tag "spring" as an example to explain how we obtain the candidate tags. For the tag "spring", its "ExcerptPostId" is 3625321, which means in the tags.xml, the metadata of tag "spring" is related to the row id "3625321". In this row, since the "body" of "spring" contains the keyword "Java", we take the tag "spring" as a candidate tag.

2) Obtaining final tags from the candidate tags

Candidate tags whose body contain the keyword of "Java" are not necessarily strongly stand for Java. Therefore, we define two indicators as H1 and H2. H1 indicates the correlation between a tag and the tag of "Java". If we want to calculate H1 of a tag A, we calculate by this way: H1 = Number of all question posts whose tags contain both tag A and "Java"/ Number of all question posts whose tags contain tag A. The value of H1 ranges from 0 to 1, the larger H1 is, the more relevant tag A and tag "Java" are. If the value of H1 is 1, it means that tag A only appears together with "Java". H1 shows the tightness of tag A and the tag "Java", but using H1 to stand for Java topic has some limitations. Suppose that a tag only appears in one post of the whole dataset and the post happens to be related to "Java". In this case, the tag is so specific to a question, but it is not representative to "Java", although its value of H1 equals 1. Therefore, we introduce H2 to solve this limitation, H2 indicates whether a tag A is representative to Java topic. Its calculation formula is H2 = Number of all question posts whose tags contain both tag A and "Java"/ Number of all question posts whose tags contain "Java". By Setting H1 and H2, we can obtain final tags, which are both strongly relate and representative to Java. For example, we set H1 to 0.1 and H2 to 0.001, which means a tag that appears together with "Java" in less than 10% of all the questions whose tags contain this tag will be removed. Furthermore, a tag that appears in less than 0.1% of all the questions whose tags contain "Java" will be removed. In this study, we set H1 to 0.1 and H2 to 0.001. Finally, 114 tags are collected.

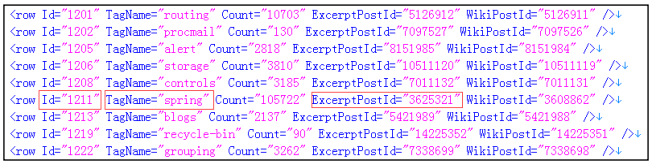

Figure 2. The metadata of "spring" tag in tags.xml.

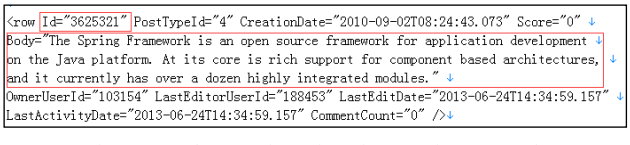

Figure 3. The metadata of "spring" tag in posts.xml.

For example, when calculating the value of tag "spring", we first calculate the parameter "a", "b", and "c" of "spring" according to the file posts.xml. For these parameters, "a" is the number of posts in posts.xml which is labeled with both tag "spring" and tag "Java", and "b" and "c" are the number of posts that contains tag "spring" and tag "Java" respectively. In our data sets, "a" is 62,101, "b" is 105,715, and "c" is 1,223,171. Based on these parameters, we get the value of H1 and H2, namely H1=a/b=0.5874 and H2=a/c=0.0508. Since H1>Thre1 and H2>Thre2, we take tag "spring" as one of the final tags.

After achieving the final tags, we manually check the final tags and delete all the tags related to Java keywords and reserved words, as these words are not topics for developers to discuss. At last, 93 tags are reserved in the final tags. These tags can be viewed as representatives for Java topics. In the following parts, we use these tags to retrieve Java related posts.

3) Finding Java-related posts

We select Java-related question posts by matching tags of post and final tags obtained in sub-step 2. Finally, 900,968 question posts are collected. In this study, we mainly use these question posts and their corresponding answer posts to make analysis.

B. Data Preprocessing

Java-related posts are collected in subsection3.1, and the content of these posts are important for further research. Therefore, to build a corpus of these text with less noise, we need preprocess the texts as follow.

1) Removing code snippets, pre snippets and link snippets. We remove such snippets, since they do not help topic models [2]. These snippets are easily identified, because they have obvious characteristics, such as, code snippets are enclosed in <code> tag, pre snippets are enclosed in <pre> tag, and link snippets are enclosed in <a> tag.

2) Removing HTML tags. These tags including and and HTML character entities etc. Since they will cause noise to our model, we need to remove them.

3) Removing stop words, numbers, and non-alphanumeric characters.

4) Stemming and Lemmatization. We use the Snowball stemmers to transform the remaining terms to their root forms (e.g. "reading" and "reads" are reduced to "read") in order to reduce the feature dimensions.

After the above preprocessing, we can get the stemming forms of all the text. In order to make the text less noisy, we will filter out stems with frequency less than 10 times, or frequency higher than 25 percent of all stem frequencies. Finally, we achieved 45,238 different words.

IV. TOPIC MODELING USING LDA

In this research, we use topic model to detect topics related to Java. LDA is a classic topic model which was implemented by Matlab Topic Modeling Toolbox 1.4 (http://psiexp.ss.uci.edu/research/programs_data/toolbox.htm). The first parameter to consider in the model is the number of topics K. In the study, we will list all the topics that the

topic model works out, and analyze the popularity and difficulty of each topic, so the value of K should not be too large. On the contrary, if the value of K is too small, it cannot distinguish all of topics about Java. Therefore, we set the value of K to be 40. There are two other modeling parameters, the hyperparameters α and β. We set α=50/K and β=0.01.

V. CASE STUDY

In the above, we present three interesting research questions, which mainly focus on topics related to Java, such as which topics are the most popular, and which ones are the most difficult to answer. In this section, we will use experimental results to answer these research questions.

A. Results of RQ1

RQ1. What topics related to Java are asked on Stack Overflow?

Motivation. Stack Overflow contains millions of posts related to Java, which imply a lot of knowledge that support Java development. In order to have a general understanding of Java, finding topics covered by these posts is significant, which can give a deep insight into Java development and advancement.

Approach. In this RQ, we use the topic distribution of a document calculated by LDA to analyze the topic of each document. LDA models a document as two metrics, namely a topic×word matrix and a topic×doc matrix. For the topic×word matrix, we can find out the key terms belonging to each topic. For the topic×doc matrix, the topic distribution of each document can be retrieved. Based on these two metrics, we can calculate the correlation coefficient of topics and documents, i.e., the questions of the Stack Overflow posts. Then, we rank the documents regarding a specific topic according to the correlation coefficient. As a result, we can understand what questions are frequently discussed for a specific topic.

TABLE II. TOPIC NAMES AND RELATED TOP 10 KEY TERMS FOR 40 TOPICS

No.	Topic Name	Key Terms (After Stemming)
1	Log	log output debug consol print warn log4j info php mode
2	Unit Test	test case junit fail run integr execut unit mock testng
3	Error	org apach core lang intern invok sun util springframework Javax
4	Java Message	messag client thread send receiv process server side queue connect
5	JavaFX	help load appreci main Javafx great advanc start amp simpl
6	Java Mail	question find link answer help solut don exampl post understand
7	Exception	except issu null throw fine occur stack wrong solv trace
8	Java Virtual Machine	time number second memori perform long size day process take
9	Cache Transaction	updat save remov will delet transact cach state manag entri
10	Web Service	xml servic generat web rest bind jaxb ws jersey jax
11	Maven Build	maven plugin depend version modul pom repositori build instal releas
12	Dependency Injection	bean configur annot context xml config inject boot support defin
13	Call Function	call function instanc insid will static don initi understand doesn
14	Java Archive	jar folder path directori jboss resourc war locat lib copi
15	Graphical User Interface	button click event gui swing display window tab close press
16	System Platform	program window instal netbean compil run system command applet version
17	Data Table	data tabl column row model record order group retriev sort
18	Mobile Development	android activ app gradl download develop sdk devic studio nativ
18	Web Page	jsp form jsf tag action compon html Javascript attribut templat
20	Programming Solution	best will solut approach better multipl good singl separ design
21	Game Development	imag screen panel game move jpanel color background size draw
22	Integrated Development Environment	view option open develop featur ui tool idea editor ide
23	HTML Parsing	text content input header html charact css replac languag format
24	Text Retrieval	field search document solr product match exampl lucen find index
25	Play Framework	ve play framework thing scala simpl doesn don figur pretti
26	Version Control	chang edit version pdf modifi current git origin doesn don
27	Stream	read write stream node differ exampl will understand tree direct
28	Java Persistence	hibern entiti map key jpa persist tabl queri parent child
29	Database Connectivity	databas connect sql jdbc mysql db insert statement store oracl
30	User Security	user access session secur login authent password usernam redirect token
31	Import Sources	add librari sourc import include packag ad exist refer find
32	Web Development	request control web tomcat servlet url http mvc respons jsp
33	Project Build	build jenkin execut job intellij script ant task command step
34	Java Generic	type implement generic interfac express lambda compil argument extend declar
35	Application Deploying	server deploy start local ejb glassfish remot run bundl contain
36	Google App Engine	app googl api engin upload python gae site appengin cloud
37	List	list select variabl valu item element dynam check loop box
38	Set Property	set properti custom default valid true enabl disabl specif fals
39	Format Conversion	object string json convert array pars format return jackson serial
40	Reporting Engine	return queri paramet pass result report filter name exampl correct

Results. We find 40 topics related to Java on Stack Overflow, to each topic, we give related top 10 key terms. Based on the key terms, we conclude a topic name for each topic. From the Table 2, we can find that topics cover a wide

range to Java, for example, topic related to Java language such as "List", "Exception" and "Call Function" etc. Topic related to version control such as "Maven Build", and "Version Control".

When calculating the correlation coefficient of a topic and a question, a question is represented as a topic distribution in LDA. Hence, it is hard to say that one question exactly belongs to a specific topic. To handle such cases, we propose the concept of the "number" of questions related to a specific topic, which is defined as the sum of correlation coefficient of all questions and a topic. Specifically, given a topic, we iterate all the questions and sum the value of topic distribution regarding this topic with the following formula:

$$\boldsymbol{Num_j} = \sum_{\boldsymbol{i=1}}^{\boldsymbol{n}} \boldsymbol{\theta_{ij}} \qquad (1)$$

where, n is the number of questions, j is the index of the topics, θ means the topic×doc matrix calculated by LDA. Each row in θ means the correlation coefficient of a topic and all the questions.

In Fig. 4, we present the "number" of the questions belonging to each topic. The x-axis is the topic ID and y-axis is the number of questions related to each topic. As shown in Fig. 4, the "numbers" of questions related to these topics are quite different. The topic with the largest number of questions is topic 21 "Game Development", and the topic with the least number of questions is topic 3 "Error". The above results show that "Game Development" has been a hot topic in Java developing community, especially after the prevalence of Android APPs.

Summary. Developers frequently discuss questions in Stack Overflow. Questions in Stack Overflow cover a wide range of Java related topics. The prevalence of these topics can be represented by the number of questions. The topic with the largest "number" of questions is "Game Development", and the topic with the least "number" of questions is "Error".

B. *Results of RQ2*

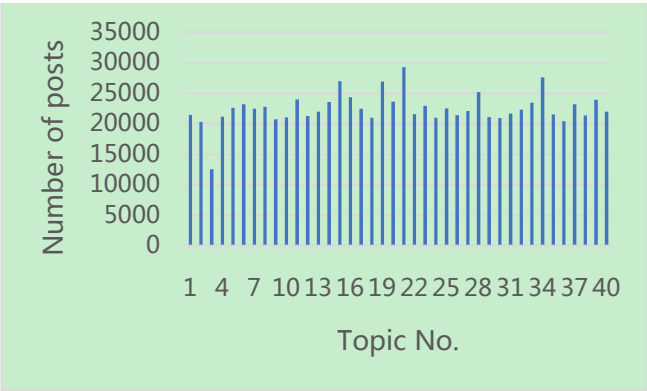

Figure 4. Statistics for each topic of questions.

RQ2. Which topics are the most popular among the questions related to Java?

Motivation. From RQ1, there are 40 topics related to Java. However, it is not enough to know the number of Java-related topics. We will investigate which topics related to Java are the most popular ones. RQ2 can help us better understand a general trend about the questions related to Java.

Approach. The main idea is to use the popularity of posts to determine the popularity of the topic. Specifically, the method will be divided into the following two steps.

1) Metrics deter *mining the popularity of post.*

In Stack Overflow. Each post contains several evaluation indicators that can measure the popularity of a post, such as "ViewCount", "AnswerCount", "FavoriteCount", and "Score", where "ViewCount" means the total number of views for the post, "AnswerCount" is Number of answers for the post, "FavoriteCount" is number of people who like the post, and "score " is average score by the viewers for the post. These four metrics can by directly obtained which provided by Stack Overflow Data Dump.

2) Obtaining post popularity by popularity of post matrix

Based on the evaluation metrics of questions, we defined the metrics of topics as follows:

$$\mathbf{metric\_topic}_j = \sum_{i=1}^{n} \boldsymbol{\theta_{ij}} \times \boldsymbol{metric\_que}_i / \sum_{i=1}^{n} \boldsymbol{\theta_{ij}} \qquad (2)$$

where n is the number of questions, j is the index of the topics, θ means the topic×doc matrix calculated by LDA, metric_que is the evaluation indicator of each question such as "ViewCount", "AnswerCount", "FavoriteCount" and "Score", metric_topic is the popularity evaluation indicator of each topics such as "Avg. ViewCount", "Avg, AnswerCount", " Avg. FavoriteCount" and "Avg. Score" in Table 3. Each row in θ means the correlation coefficient of a topic and all the questions. In contrast to the formula in section 5.1, we calculate the weighted average number of questions related to each topic, where the weight is represented by the element in the topic×doc matrix. As explained before, each post contains several evaluation indicators. Among these evaluation metrics, "ViewCount" is an ideal metrics to calculate the popularity of a post, as it measures the number of developers browsing a question. Hence, we use the average "ViewCount" as the main popularity evaluation metric. Intuitively, the more popular a question is, the higher "ViewCount" the question should be. We should note that, besides "ViewCount", other metrics can also be used to estimate the popularity of the topics. For instance, the bigger the average number of comments / favorites / score is, the more popular a topic should be.

Results. In Table 3, we list the values of all the the four evaluation metrics related to each topic. As shown in Table 3, we can see that the average "ViewCount" of most topics are larger than 1,500 times, which indicates that developers frequently browse the Java related topics in Stack Overflow.

Among these topics, "System Platform", "Java Archive", "Import Sources", "Maven Build" are the top four most popular topics. "System Platform" is related to the installation and usage of Java development platform. "Java Archive" is a widely used file format in Java that pack and

compress Java project. “Import Sources” is a common operation during developing Java projects. “Maven Build” is also a convenient step to solve Java package dependency. These topics are frequently discussed as they are common steps for developers to develop and deploy Java projects.

TABLE III. POPULARITY OF EACH TOPIC

Topic Name	Avg. ViewCount	Avg. Comment Count	Avg. FavoriteCount	Avg. Score
System Platform	2672.41	1.95	0.57	2.11
Java Archive	2468.91	1.70	0.52	1.86
Import Sources	2270.14	1.74	0.58	2.12
Maven Build	2252.81	1.59	0.61	2.24
Error	2240.27	1.78	0.45	1.57
Integrated Development Environment	2219.89	1.59	0.69	2.44
Format Conversion	2189.54	1.84	0.57	2.06
Set Property	2142.39	1.58	0.55	2.00
Dependency Injection	2141.71	1.68	0.60	1.93
Mobile Development	2138.67	1.78	0.58	1.99
Web Development	2128.16	1.74	0.51	1.51
Database Connectivity	2110.38	1.87	0.46	1.45
Log	2100.28	1.80	0.53	1.95
Stream	2085.52	1.80	0.75	2.47
Java Mail	2068.19	1.82	0.61	1.90
Web Service	2064.14	1.52	0.50	1.69
Web Page	2033.10	1.78	0.44	1.28
Programming Solution	1995.43	1.66	0.85	2.56
Java Persistence	1990.88	1.59	0.53	1.70
Unit Test	1966.97	1.68	0.54	2.10
Play Framework	1962.89	1.72	0.68	2.29
List	1956.26	1.85	0.45	1.66
Version Control	1949.87	1.81	0.56	2.03
Java Virtual Machine	1942.88	1.96	0.65	2.20
Exception	1923.35	2.07	0.42	1.59
Cache Transaction	1909.71	1.82	0.54	1.89
Java Generic	1884.67	2.03	0.76	3.02
Application Deploying	1882.27	1.51	0.50	1.54
Project Build	1880.98	1.44	0.53	1.98
HTML Parsing	1860.07	1.83	0.48	1.79
Reporting Engine	1840.69	1.74	0.49	1.73
Graphical User Interface	1797.74	1.99	0.36	1.37
JavaFX	1750.56	1.86	0.40	1.34
Call Function	1746.58	1.91	0.55	1.99
User Security	1704.65	1.60	0.50	1.46
Data Table	1646.15	1.80	0.37	1.25
Java Message	1556.15	1.53	0.47	1.51
Text Retrieval	1547.86	1.47	0.53	1.72
Game Development	1537.52	1.99	0.34	1.28
Google App Engine	1362.01	1.49	0.56	1.70

Summary. We find the top four popular topics are “System Platform”, “Java Archive”, “Import Sources” and “Maven Build”, which are the common step when develop Java projects.

C. Results of RQ3

RQ3. Which topics related to Java are the most difficult to answer?

Motivation. In this research question, we study which topics related to Java are the most difficult to answer. The topic difficult to answer should be more concerned. Particularly, if a topic is both popular and difficult, the topic should receive much more attention.

Approach. The main idea of our method is using the difficulty of posts to determine the difficulty of the topic. Specifically, the method will be divided into the following two steps.

1) Metrics determining the difficulty of post.

In Stack Overflow. If a question is difficulty to answer, there are two metrics, one is span time. Span time means the time between a created time of a post and the first accepted time of its answer. The created time of a post and the first accepted time of its answer can by directly obtained by Stack Overflow Data Dump. Another is that how many viewers can answer the question, if a question have many viewers but little answer, it means this is a difficult post. These two metrics can be used to measure the difficulty of a post.

2) Obtaining post difficulty by difficulty of post matrix.

First, we calculate the post difficulty with the time span between a question and its accepted answer with the following formula:

$$\mathbf{time\_topic}_j = \sum_{i=1}^{n} \boldsymbol{\sigma}_{ij} \times \boldsymbol{time\_que}_i / \sum_{i=1}^{n} \boldsymbol{\sigma}_{ij} \qquad (3)$$

Since we only take questions with accepted answer in consideration for analysis, when calculating the post difficulty, we only compute the time span of a question which has an accepted answer. Specifically, we examine all the questions in the topic×doc matrix. If there is no accepted answer for a question, we reset the cells of the row related to this question to 0. By this way, we get a modified matrix, which is represented with σ in the formula.

Besides the time span formula, we also measure of a question with its “AnswerCount” and “ViewCount”. Intuitively, a difficulty question tends to have a large number of views but a small number of answers. It means that although many developers are interested in this question, only a few developers are able to answer it. Thus, we use M2 to measure the difficulty of a topic:

$$\mathbf{M2}_j = \sum_{i=1}^{n} \boldsymbol{\theta}_{ij} \times \frac{\boldsymbol{ans\_cou\_que}_i}{\boldsymbol{view\_cou\_que}_i} / \sum_{i=1}^{n} \boldsymbol{\theta}_{ij} \times \mathbf{100\%} \quad (4)$$

For this metric, we first obtain the number of answers "AnswerCount" and the number of views "ViewCount" of each question respectively and the post difficulty of a question is defined as the ratio of "AnswerCount" and "ViewCount". For this formula, the smaller M2 is, the more difficult a question tends to.

Results. We list the average time span, the average answer count, and M2 score for the 40 topics in Table 4. We sort 40 topics in descending order according to the average time. We find that the top four topics are "List", "Web Development", "JavaFX" and "User Security", which need over half a month to receive an accept answer on average. On the contrary, the topics about "Java Message", "Play Framework" and "Call Function" are relatively easy to answer, which need less than 8 days to receive an accept answer on average. The metric M2 present roughly growth trend while the average time span in descending order, which means the two metrics are consistent.

TABLE IV. POPULARITY OF EACH TOPIC

Topic Name	Average Time Span (Days)	Average Answer Count	M2 (%)
List	15.83	1.39	0.76
Web Development	15.57	1.42	0.72
JavaFX	15.40	1.29	0.71
User Security	15.09	1.30	0.62
Web Service	14.50	1.33	0.69
Data Table	13.71	1.46	0.73
Java Archive	13.64	1.35	0.75
Mobile Development	13.56	1.48	0.60
Import Sources	13.34	1.39	0.56
Programming Solution	13.31	1.27	0.66
System Platform	13.30	1.36	0.71
Text Retrieval	12.46	1.33	0.70
Cache Transaction	12.42	1.46	0.77
Stream	12.19	1.39	0.69
Reporting Engine	11.84	1.42	0.76
Set Property	11.81	1.49	0.78
Maven Build	11.78	1.33	0.68
Exception	11.65	1.44	0.73
Database Connectivity	11.53	1.28	0.63
Unit Test	11.49	1.57	0.78
Dependency Injection	11.47	1.36	0.75
Java Virtual Machine	11.36	1.47	0.80
HTML Parsing	11.33	1.41	0.70
Format Conversion	11.27	1.34	0.67
Version Control	11.19	1.49	0.78
Graphical User Interface	11.04	1.38	0.64
Project Build	10.52	1.51	0.71
Integrated Development Environment	10.40	1.33	0.77
Java Generic	10.05	1.38	0.80
Application Deploying	9.92	1.30	0.74
Error	9.85	1.43	0.74
Google App Engine	9.42	1.33	0.80
Java Persistence	9.38	1.32	0.57
Log	9.05	1.72	0.84
Web Page	8.29	1.30	0.80
Java Mail	8.12	1.36	0.83
Game Development	8.08	1.47	0.82
Java Message	7.54	1.37	0.85
Play Framework	7.54	1.49	0.85
Call Function	7.23	1.39	0.96

Summary. Different topics vary a lot in the average time span for an accepted answer. Questions about "List", "Web Development", "JavaFX" and "User Security" are the top four most difficult questions. In contrast, questions about "Java Message", "Play Framework" and "Call Function" are relatively easy to answer. The average of time span of a post and the matrix M2 has a roughly similar tends. When a post is difficulty, it usually has few answers and the time span between the question and the accepted answer is large.

VI. CONCLUSIONS

In this paper, we conduct a large-scale study on Java-related posts on Stack Overflow. We first extract posts related to Java by tags of posts. Then, we use LDA, a classic topic model, to find topics among millions posts related to Java. Next, we study the most popular topic related to Java according to one major metric (i.e., the average number of views), and three minor metrics (i.e., the average number of comments, the average number of favorites, and the average score). Furthermore, we investigate the most difficult topic to answer by two metrics (i.e., the average time needed for an accepted answer and the average proportion of the number of answers to the number of views). Finally, 40 topics are found, and give abundant analysis to the most popular topic and the most difficult topic, which can give general understanding of Java.

REFERENCES

[1] *L Ponzanelli, A Mocci, A Bacchelli, M Lanza. Improving Low Quality Stack Overflow Post Detection [J].IEEE International Conference on Software Maintenance & Evolution ,2014:541-544.*

[2] *Anton Barua, Stephen W. Thomas, Ahmed E. Hassan.What are developers talking about? An analysis of topics and trends in Stack Overflow[J].Empirical software engineering,2014,(3):619-654.*

[3] *L Ponzanelli, G Bavota, MD Penta, R Oliveto, M Lanza. Prompter: Turning the IDE into a self-confident programming assistant [J].Empirical Software Engineering ,2015,21(5):1-42.*

[4] *C Treude, O Barzilay, MA Storey. How do programmers ask and answer questions on the web? (NIER track) [J].International Conference on Software Engineering ,2011,111(2):804-807.*

[5] *PS Kochhar. Mining Testing Questions on Stack Overflow [J].International Workshop on Software Mining ,2016:32-38.*

[6] D Ye, Z Xing, CY Foo, QA Zi, J Li, N Kapre. Software-Specific Named Entity Recognition in Software Engineering Social Content [J].IEEE International Conference on Software Analysis ,2016:90-101.

[7] L Mamykina, B Manoim, M Mittal, G Hripcsak. Design Lessons from the Fastest Q&A Site in the West [J].Sigchi Conference on Human Factors in Computing Systems ,2011,59(2):2857-2866.

[8] Xin-Li Yang, David Lo, Xin Xia, Zhi-Yuan Wan, Jian-Ling Sun, What security questions do developers ask? A large-scale study of stack overflow posts. Journal of Computer Science and Technology 31(5): 910–924 Sept. 2016. DOI 10.1007/s11390-016-1672-0

[9] Christoffer Rosen and Emad Shihab, What are mobile developers asking about? A large scale study using stack overflow. Empir Software Eng (2016) 21:1192–1223 DOI 10.1007/s10664-015-9379-3

[10] DM Blei, AY Ng, MI Jordan. Latent Dirichlet Allocation [J].The Journal of Machine Learning Research Archive ,2003,3:993-1022