On the Impact of Social Network Information Diversity on End-User Programming Productivity: A Foraging-Theoretic Study

Xiaoyu Jin
Department of Electrical Engineering and Computing Systems
University of Cincinnati
Cincinnati, OH, USA
jinxu@mail.uc.edu

Nan Niu
Department of Electrical Engineering and Computing Systems
University of Cincinnati
Cincinnati, OH, USA
nan.niu@uc.edu

Michael Wagner
Division of Biomedical Informatics
Cincinnati Children's Hospital Medical Center
Cincinnati, OH, USA
michael.wagner@cchmc.org

ABSTRACT
Social network information (SNI) plays an important role in providing hints to help facilitate daily software engineering tasks, especially for end-user programmers. Social information foraging theory quantitatively predicts the effect of diversity of hints on productivity. In this paper, we explore how to best leverage this theoretical prediction to support software change tasks. Specifically, we analyze the data from an observational study involving 20 bioinformatics researchers using SNI to solve software change tasks. We further classify the SNI support by using 5 diversity categories: social network type (e.g., wiki, Q&A, etc.), contributor role (e.g., core, marginal, etc.), number of contributors, information needs concerning software architecture, and information needs organized by complexity. Our results show that the contributor role best manifests the hint diversity, and its incorporation with architectural considerations could further improve productivity. Our research offers principled guidelines for supporting better use of SNI in end-user programming.

CCS Concepts
• Human-centered computing~Social networks • Human-centered computing~Collaborative and social computing systems and tools

Keywords
End-user programming; diversity of hints; social network information; social information foraging theory;

1. INTRODUCTION
Today's generation of software developers, especially end-user developers [9], frequently uses social network information (SNI) to solve their programming tasks [4]. At least two factors affect the end-user programmers' SNI usage efficiency. One is the users' individual differences in obtaining useful information [17]; another is how the SNI is structured to serve for an easy and fast searching process [3]. In order to best organize SNI to support developers' problem solving during programming, studying how the SNI is utilized in software engineering is a prerequisite.

Previous efforts have highlighted the impact of social media for enabling new ways for software teams to form and work together [5, 26]. The study by Vasilescu *et al*. described that users and developers in Q&A websites exhibited different behaviors [35]. The content of developers' blogs was analyzed by Pagano and Maalej, showing that the most popular topics represented high-level concepts whereas source code related topics were covered in less than 15% of the posts [28]. The wikis would be much better used to produce, rather than merely store, a project's documentation since project member could make a contribution [28]. These studies show how various social media are being used and their respective advantages. However, the focuses were mainly on an individual kind of SNI in general. The use of SNI during a programming task is complex and will likely involve different SNI kinds and their interactions. This paper takes into account the various SNI kinds during software change tasks.

To tackle information-intensive change tasks in software engineering, Pirolli's information foraging theory [31] attracts much attention lately. The theory leverages our animal ancestors' "built-in" food-foraging mechanisms [34] to understand human's rational information seeking and gathering behaviors. Extending the theory from solo levels, Pirolli presented elementary constructs and principles of social information foraging theory [30]. These serve as our basis for studying the impact of SNI diversity on developer productivity.

Specifically, social information foraging theory predicts, in a quantitative manner, the diversity of hints in assisting in an information forager's rate of gain [32]. In this paper, we intend to confront such a mathematical quantification with the empirical data collected from an observational study. The study involved 20 bioinformatics researchers carrying out a software change task. In our analysis, we map the SNI used by our end-user programmers to hints and further operationalize 5 diversity categories to group the SNI. The rate of gain is characterized in our work based on the progress made in terms of completing the given software change task. We aim to examine which SNI diversity operationalization best fits foraging theory's prediction, and how to combine SNI hints to improve end-user developers' productivity.

The contributions of this paper lie in the novel perspective that frames developers' usage of SNI with the social information

DOI: http://dx.doi.org/10.1145/12345.67890

foraging model, and the statistical analysis that confronts the theoretical model with empirical data. Our work provides not only a theoretical foundation for understanding developers' information seeking in the use of the SNI, but also a practical means of comparing and evaluating SNI categorization methods.

2. BACKGROUND AND RELATED WORK

Information foraging theory was originally inspired by appeals in the psychology literature for an ecological approach to understanding human information-gathering and sense-making [31]. Pirolli [31] laid out the basic analogies between food foraging and information seeking on the Web: predator (human in need of information) forages for prey (the information itself) along patches of resources and decides on a diet (what information to consume and what to ignore). Inspired by human's adaptive interaction with information on the Web, researchers began to apply foraging theory in software engineering. Notably, Lawrance *et al.* [18, 19] have made tremendous strides in understanding programmer navigation in debugging by viewing programmer as predator and bug-fix as prey. Our work expanded the applicability of foraging theory in software engineering [6, 23, 24, 25]. While our work investigates factors (SNI hints) and their interactions, we exploit directly to the social information foraging theory's prediction which we discuss next.

The applications of foraging theory in software engineering so far have mainly focused at an individual level. However, today's software is rarely developed by soloists but is the result of collective action. Pirolli has extended information foraging theory to the social level [30]. This multilevel model is derived from the quantitative theories of cooperative problem solving [13] and foraging in social groups [10]. The key assumption connecting between the solo-level foraging and the social-level is shared set of *hints*, which provides likely location of useful information that will yield some amount of utility for one or more foragers [30].

Hints, for example, can be in the form of tags in social tagging [36] systems or in software development environments [30]. Shared tags, contributed by individuals, provide navigation paths (hints) to available content that potentially improve information search. Building on the "tags-as-hints" instantiation [30], we posit that many types of individual contributions (e.g., blogs, wiki edits, question answers, etc.) can also be treated as hints, which if shared, will offer varying amounts of utility to a developer's information foraging.

Such an effect is quantitatively described in [30]. Figure 1 shows the theoretical prediction. It is the diversity of hints (*H*), rather than the hints themselves, that is predicted to impact an forager's cumulative rate of gain (*R*). Hint diversity is important because hints may vary in the validity of the search information conveyed, may vary in how they are interpreted by the information forager, and may vary in effectiveness depending on the timing they are exchanged in the search process [30]. For instance, to the extent that hints may contain correlated or even redundant search information, the effectiveness of hints will depend on what hints have already been processed. The *H* in Figure 1, therefore, should be interpreted as the distinct kind of hints, i.e., the independent heuristic effectiveness of a given type of hints.

Because hints can overlap, their effect on foraging efficiency is not monotonically increasing. Drawing on the derivation by Huberman [13], Pirolli [30] showed that the probability density distribution for finding valuable information could be cast as a log-normal distributive function of the sample of hints contributed by individuals, as shown in Figure 1. The log-normal

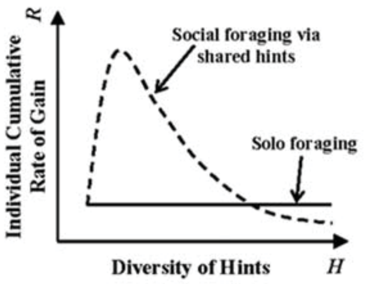


Fig. 1. Log-normal distribution of information foraging prediction. The rate of gain for solo foraging is a constant value while that for social foraging is increasing rapidly and then decreasing since the hints can overlap.

distribution makes interesting predictions about productivity of a forager receiving hints with respect to high-utility search results. If one assumes that the various states of a search space have a binomial distribution of utilities, then the search performed by the solo forager with mildly effective but not shared hints will return a distribution of result values shown by the solo-curve in Figure 1. Increasing hint diversity will shift that distribution to a log-normal and will especially increase the likelihood of search results at the higher end of the utility spectrum. As more diverse kinds of hints are received, the productivity begins to decrease due to the diminished value (e.g., redundant hints) and unneglectable cost of processing the hints.

In summary, social information foraging models like the one depicted in Figure 1 extend the theory's explanatory and predictive power to the social-level phenomena of foraging with shared hints. How good is the theoretical prediction about hint diversity on productivity when confronted with empirical study's results? This is precisely the question that drives our research.

3. STUDY DESIGN

Our research objective is to study how the diversity of SNI hints impacts end-user's productivity during software change task. To obtain end-user's SNI usage data, we performed a human subject experiment of completing a software change task.

3.1 Participants

The population that our study intends to impact are end-user developers. We select bioinformatics researchers who develop biomedical software as target group of end-user developers [1, 14]. Twenty participants took part in our experiment (12 male and 8 female; 18 graduate students and 2 staff researchers). These participants were recruited from the Cincinnati local community via email invitations. To be eligible to participate in our experiment, each individual had to consider modifying software is common and essential in their practice. Our participants had a varied background: 13 had no professional software development experience, 1 had less than a year professional experience, 3 had 1-5 years, and 3 had more than 5 years. Table 1 overviews the demographics of the participants in our study.

3.2 Task

Each participant was given 30 minutes to perform a software change task with direct biomedical relevance. The task was selected based on the intent to best simulate bioinformatics researchers' actual programming tasks. For the change task, an open-source software acted as the target system where the software change was expected to take place.

Table 1. Overview of participants and their self-reported pre-experimental survey data: "#" represents number of participants, "Area" shows participant's main research area, "PL" denotes one or more programming languages that the participant is familiar with, "SE Freq" classifies the software engineering (coding, debugging, etc.) frequency, and "Change Freq" indicates the frequency of software change task.

Area	#	PL	#	SE Freq	#	Change Freq	#
Genomics	5	Python	16	Daily	12	Frequently	11
Gene regulatory networks	4	C/C++	7	Weekly	4	Sometimes	7
Biostatistics survival analysis	4	Matlab	5	Monthly	1	Rarely	0
Neuroimaging analysis	3	R	5	Others (as needed, project-drive, etc.)	3	Others (in-house maintenance, etc.)	2
Others (molecular biology, proteomics, etc.)	4	Others (C#, Java, etc.)	14				

ImageJ [2] is a Java image processing program. We downloaded the latest version of ImageJ (v1.49) and ran it as a standalone application on a Windows lab machine. ImageJ provides extensibility via Java plug-ins. Our change task was inspired by protein quantification with ImageJ. In particular, we pre-processed an image containing a variety of different proteins being separated on a gel. We stored the pre-processing results in 4 textual files, which were provided as inputs to the software change task: Protein.txt defining the values on the x-axis and each of Result1.txt, Result2.txt, and Result3.txt giving rise to a protein sample. For each sample, the participant was asked to add a new plug-in feature so as to draw the gel plot and perform linear regression of that plot. Biologically speaking, the best protein fit is the sample with the greatest R^2 value.

4. RESULTS AND ANALYSIS

4.1 Raw Data Extraction

The computer screen was recorded as a video to capture each participant's behavior. We generated the raw data as shown in Table 2 by analyzing these videos. We split each participant's process of finishing the task into time steps based on the window environment they were working on. For example, "IDE" in Table 2 means the participant was focusing on the IDE window to program on software source code; "SNI1" means focusing on first social network information searched and accessed. We saved the actual SNI webpage links for later analysis. We recorded duration time in minutes and actions for each time step. For each time step, productivity was evaluated by the designer of the task mainly based on lines of code along with its complexity [29]. Gain was then calculated indicating the percentage of contribution this time step would contribute to the overall task completion.

4.2 Categorizing SNI

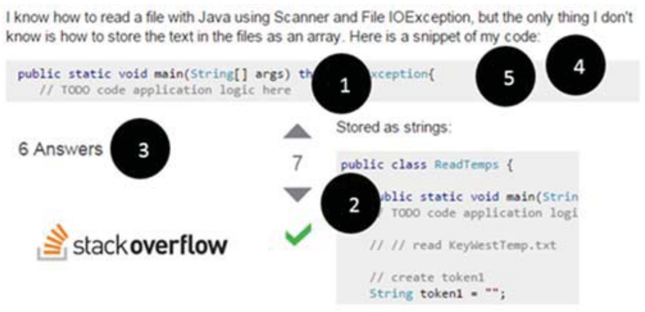
To study how the diversity of SNI hints impacts productivity, we first need to categorize the SNI hints. We consider the diversity from three perspectives. The first perspective is the format and structure of the SNI hints, which can influence the methods used by programmers to forage the information. This perspective of diversity relates to the hint diversity described by Pirolli that hints may vary in how they are interpreted by the information forager who receives them [30]. The second perspective of hint diversity is the social factor behind the SNI. Since humans are the most important resource, it is helpful for decision making if readers of a webpage know who wrote the content and how many people have posted discussions in this webpage [27]. This perspective relates to Pirolli's description that hints may vary in the validity of the search information conveyed [30]. The third perspective is about what kind of questions the SNI can answer. Example questions include: Is this SNI about the general concept of problem or related to programming technique or talking about the software architecture? Is this SNI describing a simple independent topic or a complex relation of multiple subjects? This perspective indicates hints may vary in the types of information conveyed following Pirolli's reasoning. Specifically, we utilized five diversity categories and Figure 2 illustrates the general idea utilized to classify an example SNI webpage according to the five diversity categories. Below are the five diversity categories:

Table 2. Raw data extracted from experiment video

Time step	Duration(m)	Gain	Action
IDE	8.00	17%	Editing
SNI1	0.80	3%	Scanning
SNI2	0.53	2%	Scanning
SNI3	3.58	50%	Copying
IDE	4.00	7%	Editing, debugging
SNI4	1.12	7%	Copying
IDE	6.00	3%	Editing, debugging

1) Social network type: From the form and structure of a webpage, the SNI can be categorized as wiki, Q&A (e.g., Stack Overflow), blog, social networking sites (e.g., Facebook), content communities (e.g., YouTube), etc. [15]. Our rationale is that various types of social network function differently in assisting programmers solving their problems. For example, wiki generally contains more complete and larger amount of information since it is edited by multiple approved people [11]. Therefore, wiki is better used to gain a thorough understanding of a subject, rather than to quickly find related information, which is the main strategy for opportunistic programming. On the contrary, Q&A webpages focus on a specific question. It lacks completeness, but can provide fast solution without much redundant information [37]. In our case, we followed the classification strategy and further divided these categories including wiki, Q&A, software API, software repository, and developer tutorial. Figure 2-❶ shows a Stack Overflow webpage containing a programmer's question and other programmers' answers. We therefore labeled this webpage as "Q&A".

2) Contributor role: This category reflects the importance of human factor behind the SNI. In GitHub, popular users do influence their followers by guiding them to new projects [7]. A study of three SourceForge projects shows that projects evolve from a single "hub" to a core/periphery structure [20]. This multi-layered structure was also studied by Mockus *et al.* who found that in successful open source development, a group larger by an order of magnitude than the core will repair defects, and a yet larger group will report problems [21]. Inspired by these studies, we categorized SNI as core author, main author or marginal author, which acts as an indication of the contents' reliability according to the author. Core authors are like the founders or core developers of a software program who have the highest authority about this software. Main authors like debuggers have less authority regarding the software. They are familiar with the software, but their understanding is limited. Marginal authors have the least authority since they have a shallow understanding


I know how to read a file with Java using Scanner and File IOException, but the only thing I don't know is how to store the text in the files as an array. Here is a snippet of my code:

```
public static void main(String[] args) th          xception{
   // TODO code application logic here
```

6 Answers

7

Stored as strings:

```
public class ReadTemps {

     ublic static void main(Strin
       TODO code application logi

    // // read KeyWestTemp.txt

    // create token1
    String token1 = "";
```

Fig. 2. A sample SNI website – some contents are omitted and rearranged. The numbers in black circle indicate the category type. ❶: Q&A. ❷: core author. ❸: number of contributor role. ❹: question of change implementation. ❺: question of finding focus points.

of software and can only give speculative ideas. Since contributor role is a kind of assessment of authority of one's words, it can be indicated by the number of other users' approval on the content. Figure 2-❷shows that seven users approved the usefulness of this answer meaning the content is valuable, so we treat this answer as core author hint. Here, we use five up-votes as threshold, which is a minimum threshold triggering reputation change in Stack Overflow [22]. The threshold five is not applied to all kinds of SNI. Other kinds of SNI have different thresholds or metrics.

3) Number of contributor role: The number of contributor role is another indicator of the value and quality of content. Various views probably will provide more value and nutrition than an individual statement. The idea is in line with the study by Singer *et al.* that we cannot rely on the popularity of a single user's idea in social network [33]. For an idea to gain traction in an online social network, multiple users need to post about it. However, too many people's ideas may get readers overwhelmed and lower the efficiency of finding valuable information. For instance, the topic in a forum may get thousands of posts and discussions which contain too much information. Moreover, the posts are only ordered by posting time, making it difficult to find high quality information in a short time. Here, we categorize SNIs into single author, several authors (less than 10) or many authors (more than 10). The threshold 10 we chose was based on the observation from the videos that participants maximally would read 10 posts from a website. Figure 2-❸indicates there are seven contributors to this webpage including the initial asker, so the webpage is labeled as "several authors" hint.

4) Information needs concerning software architecture: Our recent work [14] classifies the questions asked by end-user developers during the change tasks into five categories: problem understanding, problem solving, change implementation, component and structure, and software infrastructure. The idea behind the five categories is that there are two ends of questions; one is about domain concept of a problem to be solved, the other is about the architecture of the software program solving this problem. The other three categories are in between the two ends. We want to test if the diversity of SNI hints could be captured by architectural concern. Specifically, we categorized SNI by assessing which category of questions it is answering. For the example webpage in Figure2-❹, the information need is how to read file with Java and store the text as an array. The question was searched by our participant when he was in the middle of implementing a function. The search was triggered by implementation and used for implementation, thus we labeled this webpage as the "change implementation" hint.

5) Information needs organized by complexity: Sillito *et al.* [32] identified 44 specific questions programmers ask during software development and further classified those questions into four groups: (1) finding focus points, (2) expanding focus points, (3) understanding a subgraph, and (4) understanding groups of subgraphs. The categories were considered by Sillito *et al.* in terms of the amount and type of information required to answer a given question [32], which we summarized as the complexity to answer a question. We want to test how well the diversity of SNI could be captured by this categorization. In Figure 2-❺, the question is how to read file with Java and store the text as an array. It is a simple technical question with no relation to any other technical or problem domain issues, so we labeled it as the hint of "finding focus points".

4.3 Data Refinement

Previously, the raw data were generated by analyzing the videos recorded for each participant. Since our study focused on the SNI used by the participants, we needed to keep only the time steps with SNI utilized. Thus we removed data of other IDE time steps as shown in Table 3. Based on the raw data, we calculated cumulated time and gain values. Then, we divided each cumulated gain value by cumulated time to obtain the rate of gain values, which constitute the values for Y axis of Figure 1, according to the mathematical model of social information foraging theory. For instance, in Table 3, after using SNI2, the time spent on SNI was 1.33 minutes and the participant achieved 8% of task completion, thus the rate of gain was 8% ÷1.33 =0.060.

Now that we obtained the rate of gain values for Y axis, we needed to formulate values for X axis in Figure 1. In particular, were shaped the data by labeling the SNI as hint according to the five diversity categories defined previously as shown in Table 4. Then, we grouped the same type of SNIs and assigned the X axis values according to the number of SNI categories participant had already used. For each type, we assigned 1 unit X axis value to it, and divided the X axis evenly if there were several SNIs in it. For example in Table 4, for first type, there were two SNIs in it, so the first X value for SNI1 is assigned as 1/2 = 0.5 and the second one is assigned as 1.0. This strategy can be imagined as shrinking the X space whenever there are several SNIs grouped into one type.

After obtaining X values and Y values for a diversity category, we drew the fitting curve (Figure 3) to have an intuitive feeling of how the curve looks like. The curves obtained by the other four categorization strategies are overall similar to Figure 3.

According to the social information foraging model (Figure 1), the rate of gain versus hint diversity curve should follow log-normal distribution. While from Figure 3, the distribution of first two points on the left side is nothing like log-normal distribution. We found that there was a "cold start" problem. The 20 participants had a quick scan of one or two SNI links that were not directly useful to the task solving. The participants used this kind of webpages to warm up with the task and help them get into the task and we call it warm up SNI. Our participants averagely scanned 1.8 warm up SNI links before they got into the real task solving. This phenomenon is in line with the Ying and Robillard's study [37], which investigated whether the nature of a change task has any relationship with when a programmer would edit code during a programming session. Their results showed that an enhancement task was less likely to be associated with a high fraction of source code edit events at the beginning of the programming session.

They also found that this could be explained by the fact that enhancement tasks might require exploration throughout the trace. Their conclusion exactly matched our observation since our change task belonged to enhancement task. Considering this point, these starting SNI webpages foraged by participants did not satisfy our study purpose since they did not contribute to completing the change task. Therefore, we removed these data from raw data to perform analysis. As indicated by the right side of dotted line in Figure 3, the curve looks much more like a log-normal distribution. This step unblocked our way to further analyze log-normal regression described in the following section.

4.4 Log-normal Regression

After the warm up SNI links were removed, our next step was to fit the five categories of data into a log-normal regression curve to see how good the fitness was and which one best fit log-normal curve. Here, we used Microsoft Excel to perform log-normal regression. This analysis step is based on the log-normal distribution formula:

$$f(x) = \frac{1}{x\sqrt{2\pi\sigma^2}} e^{-\frac{(lnx-\mu)^2}{\sqrt{2\sigma^2}}}$$

With the obtained μ and σ for the formula, we added more points and drew the curve in Figure 4, which gives intuitive feeling of how well the curve is fitted to log-normal distribution.

Similarly, we plotted five regression curves for each participant according to the five diversity categories. For each curve, a pair of μ and σ values (Table 5) was obtained to indicate the fitness of the regression. We will take one participant as example to show the results and then discuss the situations for all participants.

Table 3. Calculating rate of gain

Time step	Time accumulated(m)	Gain accumulated	Rate of gain
SNI1	0.80	4%	0.050
SNI2	1.33	8%	0.060
SNI3	4.92	70 %	0.142
SNI4	6.03	87%	0.144
SNI5	6.75	94%	0.139
SNI6	7.52	98%	0.130
SNI1	8.07	100%	0.124

Table 4. Categorizing SNI and assigning X axis values according to category #1: SNI type.

# of types of SNI	SNI category	X	Y
One type	SNI1-software API	0.5	0.050
	SNI2-software API	1.0	0.060
Two types	SNI3-source code tutorial	1.5	0.142
	SNI4-source code tutorial	2.0	0.144
Three types	SNI5-wiki	3.0	0.139
Four types	SNI6-developer tutorial	3.5	0.130
	SNI1-software API	4.0	0.124

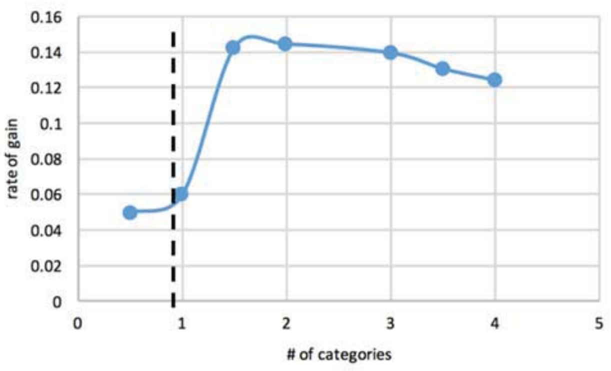


Fig. 3 Fitting data points into a curve

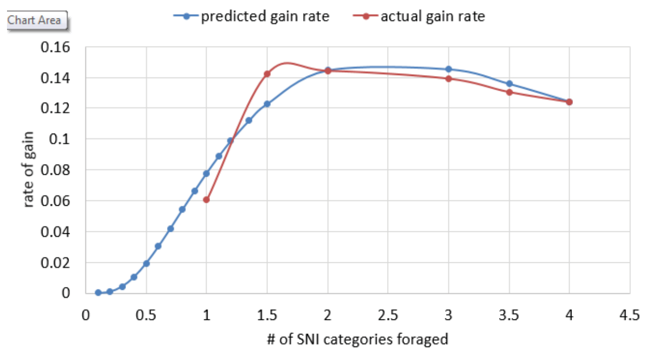


Fig. 4 Fitting log-normal regression curve

In Table 5, the category type of contributor role has the smallest value of sum of squared differences, meaning that the contributor role can best predict the productivity according to the foraging-theoretic principle. This finding applies to 15 of the 20 participants. For the five remaining end-user developers, three have the best regression fit with the information needs concerning software architecture, whereas the other two of them best fit the information needs organized by complexity. Further analysis shows that the SNIs used by the five participants have only one or two kinds of contributor roles. Hence the hint diversity based on contributor role does not exist. In situations like this, other types of diversity categorization show better ability to predict the productivity. We therefore conclude that contributor role serves as a primary factor that best manifests foraging theory's prediction. Next we analyze the interactions between the factors.

4.5 Interactions between Categories

We further studied how the interactions between two types of categorization affect theoretic prediction. Since we concluded that "contributor role" fit the best to foraging theory's prediction, we only used "contributor role" to interact with other diversity categories. Here, we used bivariate log-normal distribution [38], which basically means that the log-normal shape is a joint effect of two variables. To fit the bivariate log-normal distribution, we used a similar approach to the process of log-normal regression. The results are shown in Table 6. The difference is that now we need to adjust two pairs of variables to achieve the best fit.

Since for 15 of the 20 participants, social information foraging model best predicts the productivity with contributor role. We only analyze interactions for these 15 participants. In Table 3, when contributor role is incorporated with information needs concerning software architecture, the sum of difference value of

Table 5. Log-normal regression parameters generated

Category type	μ	σ	Sum of Diff
SNI type	1.41	0.93	6.9E-4
Contributor role	1.54	1.00	1.5E-4
# of contributor role	1.57	1.17	7.3E-4
Architecture concern	1.47	0.90	9.3E-4
Complexity concern	1.42	0.93	5.6E-4

Table 6. Bivariate log-normal regression parameters. C means the categorization type. C1: social network type, C2: contributor role, C3: number of contributors, C4: information needs concerning software architecture, C5: information needs organized by complexity.

mu (C2)	sigma(C2)	mu	sigma	Sum of Diff
6.41	8.97	C1: 6.59	C1: 10.20	1.49E-3
6.76	4.37	C3: 7.64	C3: 10.18	9.2E-4
5.58	4.46	C4:10.59	C4: 11.59	1.5E-6
6.55	7.04	C5: 15.26	C5: 21.46	6.8E-6

1.5E-6 is the smallest. This value is also smaller than the value of 1.5E-4 when contributor role is considered only, meaning that the combined effect in theoretical prediction is even better than only considering contributor role. For the 15 participants that we analyzed, when contributor role is incorporated with information needs concerning software architecture, productivity could be best predicted for 12 of them. The remaining 3 can be best predicted by incorporating contributor role with social information needs organized by complexity. Our intuition for the results is that the contributor role and information needs concerning software architecture are two relatively orthogonal directions in categorizing the SNI. Information needs organized by complexity has similarities to information needs concerning software architecture, hence sometimes it can better predict the productivity.

4.6 Threats to Validity

One of the external validity threat to our study is the representativeness of our study participants. While we tried to be inclusive, the participants were recruited from a local community and were primarily affiliated with a university's medical campus. These bioinformatics researchers may not able to represent some other kinds of end-user developers.

The definition of participants' productivity is mainly based on the experiment designer's subjective estimation of code complexity along with the lines of code generated. Such estimation may not capture the productivity objectively, which may influence the accuracy of our results. A clearer definition of productivity in our case is needed in the future work.

Currently, the metrics and thresholds used to categorize the SNI are based on our observation and subjective understanding. We need to study more literature to clearly define all these metrics and threshold values.

The removal of "cold start" points, which helped us proceed with research, is not fully proved to be a right decision. Although we have found a literature [37] discussed this phenomenon and backed up our decision, more careful study is needed to prove the decision is convincing.

5. IMPLICATIONS

We identified contributor role and information needs from architecture concern for SNI as two key factors which could potentially improve end-user developers' productivity. In addition, our strategies and metrics of labeling SNI hints can provide guidelines to develop tool support to automatically categorize the SNIs. We discuss the implications from two perspectives: SNI web designers and end-user developers.

For SNI web designers, they can design algorithms or metrics to assess the users' professional level regarding a topic. The evaluation could be done by multiple perspectives such as working background and social status. For each user, different scores can be assigned to different areas by calculating the closeness between the topic and the user's specialized area. The importance of contributor role is also discussed in previous literature. Previous studies have shown that users with high social status or high reputation tended to be more active members of a forum and they are at the core of community and have merits of their quality of contributions [8, 12]. The user level status can then be shown on the webpage to help web information forager quickly locate information from expert users. To answer what information needs a webpage can satisfy for the end-user developers, some strategies such as word frequency analysis could be utilized to summarize the main topic of a webpage.

For end-user developers, they may have not realized the importance of knowing who wrote a piece content since they care more about the content and information. However, our results which identified contributor role as an important factor to improve productivity, indicating that the chances to find useful information from user with high reputation or more expertise should be statistically higher than from general users. If the end-user developers start to realize and think about this point, they can possibly improve their efficiency when finding information considering where it comes from, especially when the webpage provides clear indication of such information.

6. CONCLUSION

In this paper, we reported the quantitative analysis on data extracted from an experiment of end-user software change task. We framed the analysis with theoretical guidance by social information foraging theory. We framed SNI diversity with five categories and analyzed which SNI diversity operationalization best fits foraging theory's prediction. We found that the contributor role best manifests the hint diversity. Moreover, its incorporation with architectural considerations could further improve the predictability of end-user programmers' productivity. The limitation is that we only performed analysis on the meta-data level of the SNI webpages while the actual contents contained in the webpages have significant impact on how foragers will use the SNI. Therefore, our future work includes further analyzing the chunks of contents on the SNI webpages.

7. ACKNOWLEDGMENTS

We thank all the participants of our study. The work is funded in part by the U.S. NSF Grant CCF-1350487.

8. REFERENCES

[1] V. Bonazzi *et al.*, "Software Discovery Index Workshop Report." https://nciphub.org/resources/885/supportingdocs, March 18, 2016.

[2] W. Rasband *et al.*, "Protein Quantification Using ImageJ." http://openwetware.org/wiki/Protein_Quantification_Using_ImageJ, March 18, 2016.

[3] E. Agichtein, E. Brill, and S. Dumais, “Improving Web Search Ranking by Incorporating User Behavior Information,” In *SIGIR*, pp. 19-26, 2006.

[4] N. Ahmadi, M. Jazayeri, F. Lelli, and A. Repenning, “Towards the Web of Applications: Incorporating End User Programming into the Web 2.0 Communities,” In *SoSEA*, pp. 9-14, 2009.

[5] A. Begel, R. DeLine, and T. Zimmermann, “Social Media for Software Engineering,” In *FoSER*, pp. 33-38, 2010.

[6] T. Bhowmik, N. Niu, W. Wang, J. C. Cheng, L. Li, and X. Cao, “Optimal Group Size for Software Change Tasks: A Social Information Foraging Perspective,” *IEEE Transactions on Cybernetics*, 46(8): 1784-1795, 2016.

[7] K. Blincoe, J. Sheoran, S. Goggins, E. Petakovic, and D. Damian, “Understanding the Popular Users: Following, Affiliation Influence and Leadership on GitHub,” *Information and Software Technology*, 70: 30-39, 2016.

[8] A. Bosu, C.S. Corley, D. Heaton, D. Chatterji, J.C. Carver, and N.A. Kraft, “Building Reputation in Stackoverflow: An Empirical Investigation,” In *MSR*, pp. 89-92, 2013.

[9] M. Burnett and B. A. Myers, “Future of End-User Software Engineering: Beyond the Silos,” In *FOSE,* pp. 201–211, 2014.

[10] L. Giraldeau and T. Caraco*, Social Foraging Theory*. Princeton Univ. Press, 2000.

[11] R. Giuffrida and Y. Dittrich, “Empirical Studies on the Use of Social Software in Global Software Development–A Systematic Mapping Study,” *Information and Software Technology*, 55(7):1143-1164, 2013.

[12] K. Hart and A. Sarma, “Perceptions of Answer Quality in an Online Technical Question and Answer Forum,” In *CHASE,* pp. 103-106, 2014.

[13] B. A. Huberman, “The Performance of Cooperative Processes,” Physica D, 42: 38–47, 1990.

[14] X. Jin, C. Khatwani, N. Niu, M. Wagner, and J. Savolainen, “Pragmatic Software Reuse in Bioinformatics: How Can Social Network Information Help?” In *ICSR*, pp. 247-264, 2016.

[15] M. Kaplan and M. Haenlein, “Users of the World, Unite! The Challenges and Opportunities of Social Media,” *Business Horizons*, 53(1): 59–68, 2010.

[16] K. S. Kim and B. Allen, “Cognitive and Task Influences on Web Searching Behavior,” *Journal of the American Society for Information Science and Technology*, 53(2): 109-119, 2002.

[17] A.J. Ko, B.A. Myers, M.J. Coblenz, and H.H. Aung, “An Exploratory Study of How Developers Seek, Relate, and Collect Relevant Information during Software Maintenance Tasks,” *IEEE Transactions on Software Engineering*, 32(12): 971-987, 2006.

[18] J. Lawrance, R. Bellamy, M. Burnett, and K. Rector, “Using Information Scent to Model the Dynamic Foraging Behavior of Programmers in Maintenance Tasks,” In *CHI*, pp. 1323-1332, 2008.

[19] J. Lawrance, M. Burnett, R. Bellamy, C. Bogart, and C. Swart. “Reactive Information Foraging for Evolving Goals,” In *CHI*, pages 25–34, 2010.

[20] Y. Long and K. Siau, “Social Network Structures in Open Source Software Development Teams,” *Journal of Database Management*, 18(2): 25-40, 2007.

[21] A. Mockus, R.T. Fielding, and J.D. Herbsleb, “Two Case Studies of Open Source Software Development: Apache and Mozilla,” *ACM Transactions on Software Engineering and Methodology*, 11(3): 309-346, 2002.

[22] D. Movshovitz-Attias, Y. Movshovitz-Attias, P. Steenkiste, and C. Faloutsos, “Analysis of the Reputation System and User Contributions on a Question Answering Website: Stackoverflow,” In *ASONAM,* pp. 886-893, 2013.

[23] N. Niu, X. Jin, Z. Niu, J. C. Cheng, L. Li, and M. Y. Kataev, “A Clustering-Based Approach to Enriching Code Foraging Environment,” *IEEE Transactions on Cybernetics*, 46(9): 1962-1973, 2016.

[24] N. Niu, A. Mahmound, and G. Bradshaw, “Information Foraging as a Foundation for Code Navigation,” In *ICSE*, pp. 816-819, 2011.

[25] N. Niu, A. Mahmound, and Z. Chen, and G. Bradshaw, “Departures from Optimality: Understanding Human Analyst’s Information Foraging in Assisted Requirements Tracing,” In *ICSE*, pp. 572-581, 2013.

[26] S. Nešic, F. Lelli, M. Jazayeri, and D. Gaševic, “Towards Efficient Document Content Sharing in Social Networks,” In *SoSEA*, pp. 1-8, 2009.

[27] N. Novielli, F. Calefato, and F. Lanubile, “Towards Discovering the Role of Emotions in Stack Overflow,” In *SSE,* pp. 33-36, 2014.

[28] D. Pagano and W. Maalej, “How Do Developers Blog? An Exploratory Study,” In *MSR*, pp. 123-132, 2011.

[29] K. Petersen, “Measuring and Predicting Software Productivity: A Systematic Map and Review.” *Information and Software Technology*, *53*(4): 317-343, 2011.

[30] P. Pirolli, “An Elementary Social Information Foraging Model,” In *CHI*, pp. 605-614, 2009.

[31] P. Pirolli, *Information Foraging Theory: Adaptive Interaction with Information,* Oxford Univ. Press, 2007.

[32] J. Sillito, G. C. Murphy, and K. De Volder. “Asking and Answering Questions during a Programming Change Task,” *IEEE Transactions on Software Engineering*, 34(4): 434-451, 2008.

[33] L. Singer, N. Seyff, and S.A. Fricker, “Online Social Networks as a Catalyst for Software and IT Innovation,” In *SSE*, pp. 1-5, 2011.

[34] D. W. Stephens and J. R. Krebs, *Foraging Theory*. Princeton Univ. Press, 1986.

[35] B. Vasilescu, A. Serebrenik, P. Devanbu, and V. Filkov, “How Social Q&A Sites are Changing Knowledge Sharing in Open Source Software Communities,” In *CSCW*, pp. 342-354, 2014.

[36] W. Wang, N. Niu, H. Liu, and Y. Wu, “Tagging in Assisted Tracing,” In *SST*, pp. 8-14, 2015.

[37] A. Ying and M. Robillard, “The Influence of the Task on Programmer Behaviour,” In *ICPC*, pp. 31–40, 2011.

[38] S. Yue, "The Bivariate Lognormal Distribution for Describing Joint Statistical Properties of a Multivariate Storm Event." *Environmetrics* 13, 8: 811-819, 2002.