Asking the Right Question in Collaborative Q&A systems

Jie Yang, Claudia Hauff, Alessandro Bozzon, Geert-Jan Houben
Delft University of Technology
Mekelweg 4, 2628 CD
Delft, The Netherlands
{j.yang-3, c.hauff, a.bozzon, g.j.p.m.houben}@tudelft.nl

ABSTRACT

Collaborative Question Answering (cQA) platforms are a very popular repository of crowd-generated knowledge. By formulating questions, users express needs that other members of the cQA community try to collaboratively satisfy. Poorly formulated questions are less likely to receive useful responses, thus hindering the overall knowledge generation process. Users are often asked to reformulate their needs, adding specific details, providing examples, or simply clarifying the context of their requests. Formulating a good question is a task that might require several interactions between the asker and other community members, thus delaying the actual answering and, possibly, decreasing the interest of the community in the issue. This paper contributes new insights to the study of cQA platforms by investigating the editing behaviour of users. We identify a number of editing actions, and provide a two-step approach for the automatic suggestion of the most likely editing actions to be performed for a newly created question. We evaluated our approach in the context of the Stack Overflow cQA system, demonstrating how, for given types of editing actions, it is possible to provide accurate reformulation suggestions.

Categories and Subject Descriptors: H.3.3 Information Storage and Retrieval: Information Search and Retrieval
General Terms: Experimentation
Keywords: Collaborative Question Answering, Classification, Stack Overflow

1. INTRODUCTION

Collaborative Question Answering (cQA) systems are highly popular Web portals where everyone can ask questions, and (self-appointed) experts jointly contribute to the creation of evolving, crowdsourced, and peer-assessed knowledge bases [5][19], often in a reliable, quick and detailed fashion. Examples of such portals are Yahoo! Answers[1] (for all kinds of questions) and Stack Exchange[2], which consists of a number of sub portals, each dedicated to a particular topic, such as travelling, mathematics or programming.

In cQA systems users (*askers*) post questions, and rely on other community members to provide a suitable solution to their information need. Potential *answerers* (users that answer questions) look through the list of existing questions, typically ordered by recency, and decide whether or not to contribute to ongoing discussions. Such decisions are influenced by a multitude of factors, including time constraints, quality and difficulty of the question, and the knowledge of the answerer. Users can often also *comment* or *vote* on existing questions and answers. Commonly, when satisfied, an *asker* can mark an answer as *accepted*, thus declaring her need satisfied. Incentives to answer are often based on gamification features of a platform, such as reputation points [3].

Although the median time until a first answer is posted in response to a question can be in the order of a few minutes (as shown for instance for Stack Overflow [16]), more and more questions [4] remain ignored or without an accepted answer. Questions are unanswered when their meaning is not clear to the community members, or when it is not possible, given the available information, to understand the nature of the problem (e.g. the source code that produces a compiling error is missing). A good question should have enough details (but not too much), enough depth (without drifting from the core subject), examples (if applicable) as well as avenues already investigated by the asker [17]. Well-formed questions attract more high-quality answers than poorly formed questions, as subject experts are more likely to help users that already put some effort into finding an answer themselves [4, 16, 21].

We focus on Stack Overflow[3], a cQA platform covering a large variety of topics related to the software development domain. Introduced in 2008, Stack Overflow features more than 5 million questions, and 10 million answers provided by more than 2 million users[4]. To manage and increase the likelihood of good and useful answers, users are provided with editing functionality, which allows the improvement of questions based on the feedback from other community members. Edits usually happen in response to comments or answers, a process which might require several interactions (asker waits for comments or answers, adapts the question, waits again,

[1] `http://answers.yahoo.com/`

[2] `http://stackexchange.com/`

[3] `http://stackoverflow.com/`

[4] These numbers are based on the Stack Overflow data released in September 2013.

.

etc.) and, ultimately, might cause the question to sink in the list of open issues.

Our work contributes a novel approach to improve the question formulation process. We envision a system that upon question submission, provides askers with feedback about the aspects of the question they need to change (improve) in order to phrase their needs in the *right* way. This in turn is more likely to attract the *right* answerers.

Here, we perform a first study to investigate the feasibility of this idea. In particular, we propose and evaluate the following two-step approach:

1. Determine whether the question is of high quality or whether it requires an *edit* (**Question Editing Prediction**).
2. When an edit is required, identify which aspect(s) of the question need(s) to be improved to turn it into a high quality question (**Edit Type Prediction**).

In the process, we address the following research questions:

- **RQ1**: To what extent are traces of question edits (and the lack of edits) indicative of well or poorly formed questions?
- **RQ2**: Given sets of properly/poorly formed questions, is it possible to automatically detect which category the question belongs to?
- **RQ3**: Is it possible to predict the type of action required to make a question "better", i.e. improve its quality?

Our results show that:

1. The need for edits is indeed indicative of a question's quality.
2. The need for a question to be edited can be predicted with high accuracy.
3. The identification of the type of required edit is much more difficult to predict: we classified edit types in three categories, and found that only one of them can be accurately predicted.

In the remainder of this paper we first briefly cover related work in Section 2. Then, in Section 3 we present our methodology and developed hypotheses. The experimental setup and the experiments are presented in Sections 4 and 5 respectively. Finally, we discuss our findings and present future work in Section 6.

2. RELATED WORK

Collaborative question answering systems have been emerging as important collective intelligence platforms. Domain specific cQA platforms such as Stack Overflow are transforming the way people share experience, create knowledge and, ultimately, contribute to the evolution of a given field [22, 23].

Several works focused on the issue of question and answers quality in cQA systems, providing a solid scientific support to the premises of our work. Burns and Kotval [6] describe thirteen dimensions that can be used to distinguish questions, including answer factuality, complexity, and depth of answer needed. Dearman and Truong [7] surveyed 135 active members of the Yahoo! Answers platform, identifying the composition of the question as one of the main factors leading to its consideration by the community. Harper et al. [10] investigated predictors of answer quality in several cQA sites, identifying as relevant dimensions the question topic, type (e.g. factual, opinion), and prior effort (i.e. the requester clearly indicated a previous attempt to solve the problem). On a higher abstraction level, an investigation into Stack Overflow identified four main types of questions [17]: *Need-To-Known*, *Debug/Corrective*, *How-To-Do-It*, and *Seeking-Different-Solution*. Recent work has also considered the evolution of user behaviour over time: Ahn et al. [1] studied whether users learn to be better question askers over time, by correlating past actions (e.g. receiving upvotes or comments, accepting answers, etc.) with the quality of the subsequent ones. Past work has also investigated the nature of unanswered questions on Stack Overflow [4, 16, 21] - two of the main reasons behind a question remaining unanswered are the lack of clarity and the lack of required information (source code, etc.).

Previous work has also focused on a variety of prediction tasks, including question difficulty prediction [9], question longevity, user expertise estimation and question recommendation. Anderson et al. [2] studied the factors that contribute to the long-lasting value of questions in Stack Overflow. Liu et al. [13] proposed a competition-based model for estimating question difficulty by leveraging pairwise comparisons between questions and users. Another area related to our work is the estimation of user expertise in cQA systems. In [24] it was found that the expertise networks in cQA systems possess different characteristics from traditional social networks, and based on this finding an expertise metric was proposed. Similar aspects were also studied in [12, 19]. Relevant examples of contributions addressing the problem of routing questions to the right answerer can be found in [14, 15] and [25].

To the best of our knowledge, no previous work has targeted the problem of question editing in cQA systems. Iba et al. [11] analysed editing patterns of Wikipedia contributors using dynamic social network analysis; although several observations are related to our setting, the nature and purpose of wikis is different from the one of cQAs. The type and nature of collaborative acts was studied in [20] on the specific example of users proposing novel mathematical problems, or contributing to their solutions. While providing important insights, [20] focused on a qualitative assessment of the collaboration problem. The application of those insights, e.g. by means of automatic analysis methods, was not investigated.

3. METHODOLOGY

This section describes our experimental methodology. We first discuss and present the types of question edits typically encountered on Stack Overflow. Publicly available data dumps[5] contain the entire history of all questions posted to Stack Overflow. Every revision of a question includes information about the editor (the asker or another user) and the time of the edit. We considered only questions whose question body was edited, thus ignoring changes in the title or in the tags.

[5] `https://archive.org/details/stackexchange`

Then, we discuss how we approached the *edit prediction task* as well as the *edit type prediction task* (Section 3.2). Finally, Section 3.3 presents a number of hypotheses, derived from our research questions of Section 1.

3.1 Common Question Edits

We first need to define when we consider a question to be of high and of low quality respectively.

A question is of high quality and thus **well formed** if:

1. it has not been edited in the past; and,
2. it has received at least two answers (the median number of answers for questions on Stack Overflow).

Previous work [18] relies on the number of positive preferences (upvotes) as question quality indicator. Due to the significant correlation between upvotes and number of answers[6] we settled on the number of answers as indicator.

In contrast, we hypothesise that a question might be initially of **poor quality** if it does not receive an answer within 12 minutes after its publication (the median answer time on Stack Overflow), or if it is edited one or more times before it receives the first answer.

However, not all edits are equal: a question may be edited by the asker herself or by a different Stack Overflow user[7]; an edit can lead to a major change in semantics or be simply a correction of a spelling error or a re-formatting of the question.

In order to gain qualitative insights, we first conducted a small-scale study aimed at eliciting the most important edit categories on Stack Overflow. We define as *important* the first edit (in the sequence of edits) that is temporally followed by one or more answers.

We randomly selected 600 (*question,important edit*) pairs, and had three trusted annotators describing the nature of the observed changes. We found that most of our edits fall into one (or more) of the following eight categories:

- **Source code refinement**: the provided source code is modified; additions are more frequent than removal or truncation.
- **Context**: the asker provides additional context and clarifies what she wants to do/achieve, as well as information about the "bigger picture" of this question.
- **HW/SW details**: inclusion of additional details about the hardware and/or software used (software version, processor specification, etc.).
- **Example**: the asker provides examples of inputs, or describes the expected results.
- **Problem statement**: the asker clarifies the technical nature of the problem by posting an error message, stack traces or log messages.
- **Attempt**: the asker details the attempts she already made in order to solve the problem, either before posing the question or in response to comments or posted answers.
- **Solution**: the asker adds/comments on the solution found for the question. The Stack Overflow community explicitly encourages contributions where the user asking the question also provides the final answer. Some askers append their solutions, others create an answer in the discussion.
- **Formatting**: the asker fixes small issues including spelling errors and code formatting.

Table 1 provides an example of each edit type found in our data set (described in detail in Section 4), apart from the *formatting* category. This initial study shows that the most important edit types are related to question clarification as well as to the description of attempts made to solve the problem - including the working solution. We therefore decided to not further consider the *formatting* category.

3.2 Predicting Edits and Edit Types

Extracting Useful Question Edits.

The purpose of this step is to create the training and test data sets for our experiments. Our goal is to create a data set characterised by the presence of two distinct classes of questions, which will be used to train a classifier able to properly identify *edited questions* from *non-edited questions*.

Edited questions were selected as follows. Let there be n edits of question Q_i expressed as revisions $R^{i_1}_{t_{a_1}}, ..., R^{i_n}_{t_{a_n}}$. Here, Q_i can also be considered as $R^{i_0}_{t_{a_0}}$, i.e. the original question posted at time t_{a_0}. Revision IDs are sorted according to time, each subsequent revision is an edit of the previous revision.

Users (the asker as well as anybody else) can also *comment* on a question or *answer* it. Let $C^i_{t_j}$ be a comment on question Q_i or any of its revisions at time t_j. Similarly, let $A^i_{t_k}$ be an answer to question Q_i (or any of its revisions) at time t_k. Which revision the comment or answer are referring to, depends on the timestamp of the comment or answer. We exploit these comments and answers and extract all pairs of original & edited question, with the following sequence characteristics:

$$R^{i_0}_{t_{a_0}} \rightarrow C^i_{t_j} \rightarrow R^{i_1}_{t_{a_1}} \rightarrow A^i_{t_k} \quad (1)$$

where $t_{a_0} < t_j < t_{a_1} < t_k$. The idea is to be able to automatically catch edits stimulated by discussions with the community.

Intuitively, we consider edits that:

- have been made potentially in response to a first comment; and
- after the edit, triggered the posting of an answer.

To further ensure that the edits occurred in response to the posted comment, we only consider those pairs of original and edited questions where there is some overlap in terms between the comment and the added text in the edit.

As an example, in response to a comment:

"Please add some source code"

a user might edit a question and add:

"My code: `[actual code]`*."*

[6]In our dataset with 5M questions, we observed a linear correlation coefficient of 0.25, p-value<0.001.

[7]Stack Overflow users are allowed to edit other users' questions after they reach a particular reputation level.

Edit Category	Post ID	Added Text (Excerpt)
Attempt (1st edit)	9943644	Update 1: I've tested the application with NHProf without much added value: NHProf shows that the executed SQL is ...
HW/SW details (1st edit)	7473762	I'm running OS 10.6.8
Source code refinement (1st edit)	13318757	Here is the code: `import android.content.Context;` `import android.graphics.Matrix;` `...`
Problem statement (1st edit)	7500461	The Error: `Exception in thread "AWT-EventQueue-0" com.google.gson.JsonParseException: The JsonDeserializer com.google.gson.DefaultTypeAdapters$CollectionTypeAdapter@4e76fba0 failed to deserialize json object ....`
Example (1st edit)	11875006	I have a list of numbers like this in PHP array, and I just want to make this list a little bit smaller. `2000: 3 6 7 11 15 17 25 36 42 43 45 ...`
Context (1st edit)	13923053	EDIT: I have 'jquery-1.8.3.min.js' included first, then I have the line $.noConflict();. Then I have includes for external files using the prototype framework. then I include my user defined function and finally call it. But, I figured ...
Solution (2nd edit)	9215463	**EDIT 2: **Okay that's done the trick. Using @Dervall 's advice I replaced the MessageBox line with a hidden window like this: `MSG msg;` `HWND hwnd;` `WNDCLASSEX wcx;`

Table 1: Each edit type example shows part of the text added in the first or second edit respectively. The *Post ID* is the Stack Overflow ID. Note that revisions of post with ID `postID` can be accessed via *http://stackoverflow.com/posts/*`postID`*/revisions*.

With this basic filtering step we were able to capture around 170K quality-enhancing edits. The resulting question-edit pairs were then ranked according to the amount of editing, measured by the number of characters changed in the edited and original version of the question.

Our *non-edited questions* were selected from among all questions that were never edited and have received at least one answer. We ranked the non-edited questions according to their number of received answers – intuitively, the more answers a question receives, the higher is the engagement of community members with the question.

Extracting Edit Types.

Based on the categories identified in Section 3.1, we conducted a follow-up annotation study on 1000 *edited* questions randomly selected from the 25K most edited questions (i.e. those with the longest edits), with the purpose to derive labelled data for our edit types classifiers.

We collected annotations[8] for the questions according to four categories derived from our initial findings presented in Section 3.1: *Code*, *SEC* (merging the categories *Problem Statement*, *Example* and *Context*), *Attempt* (merging the *Solution* and *Attempt* categories) and *Detail*. The decision to group the categories as presented was taken due to the practical difficulties the annotators encountered deciding between them. In later stages, we discarded the *Detail* category due to the small number of annotated instances. Edits which do not fall into one of our categories were labelled as a "null edit".

We note, that for every question to be annotated, *all* edits of that question were labelled, i.e. $R^i_{t_j}$ for $j = 1...n$.

The annotations were then used to train three binary classifiers aimed at providing suggestions about the type of edit to be performed, for those questions that were deemed as in need for edits.

3.3 Hypotheses

This section presents the research hypotheses, based on the research questions posed in Section 1, we investigate in our work.

- **Hypothesis 1**: Communities attracting beginner's programmers (e.g. Android programming, Web design) receive a larger number of edited questions than communities which require more in-depth knowledge

[8]We describe the annotation process in greater detail in Section 4.2.

(e.g. Assembler programming, functional programming).

- **Hypothesis 2**: Users new to Stack Overflow post questions in need of refinement. Over time, users learn how to post good quality questions.
- **Hypothesis 3**: Not only the time a user has spent on the portal is important, but also the amount of knowledge the user already has about a particular topic. We posit that users with substantial knowledge on a particular topic are less likely to post questions which require a substantial edit.
- **Hypothesis 4**: As the Stack Overflow platform gained popularity, less and less questions requiring a substantial edit have been posted. Users read the guidelines and "learn" from different forums/portals how to properly ask questions.
- **Hypothesis 5**: New users are most likely to "forget" to add source code and previous attempts to their questions.

4. EXPERIMENTAL SETUP

We use the public Stack Overflow dump[9]. Manual annotations, training and test data used in ours experiments are available for download at `https://github.com/WISDelft/WIS_HT_2014`. We consider, for training purposes, all questions posted up to and including December 31, 2012; the test set includes all questions posted between January 1, 2013 and September 6, 2013. We use a logistic regression-based classifier[10]. The feature set is composed of unigrams (terms) extracted from the dataset, an approach that has been shown to perform well for different prediction tasks in the past. The chosen classifier, though likely to not yield the best possible accuracy, allows us to gain valuable insights into the importance of different features.

4.1 Edit Prediction

The training and evaluation of the edit prediction classifier has been performed using the ranked list of edited and non-edited questions described in Section 3.2.

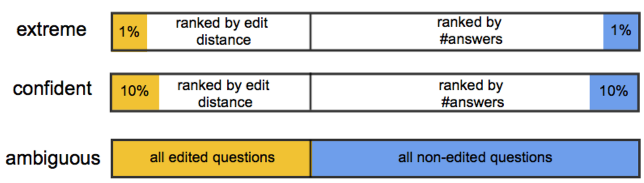

Figure 1: Both the training and test data were partitioned in three ways. The edit prediction classifier was trained on the *Extreme* set of the training data. The evaluation was performed on all data partitions of the test data.

Given these two rankings of the questions in the positive (*edited*) and negative (*non-edited*) class, we create three different data partitions, presented in Figure 1.

[9]Available online at `https://archive.org/details/stackexchange`

[10]Implemented in sklearn `http://scikit-learn.org`

- The **Extreme** set contains the top 1% of positive and negative samples.
- The **Confident** set contains the 10% highest ranked edited and non-edited questions respectively.
- The **Ambiguous** set contains all edited as well as all non-edited questions.

We derive this partitioning of the data separately for our training and test data. We train our edit prediction classifier on the *Extreme* data partition of the training data (i.e. questions posted until the end of the year 2012) and evaluate the performance of the classifier on the *Extreme*, *Confident* and *Ambiguous* data partitions of our test data (questions posted in 2013).

For training purposes, due to the skewedness of the class distribution (there are more non-edited than edited questions), we randomly sample from the negative class until we have reached the same number of samples as exist in the positive class. A similar sampling process is also used for the test data, with the exception of the *Ambiguous* set, which includes all test questions.

The reason for experimenting with different data partitions is the nature of the task. Our overall goal is to predict for each and every question in our test set whether or not it requires an edit. Due to the nature of the questions, we expect that questions in the *Extreme* test set can be classified with a higher accuracy than questions in the *Ambiguous* test set.

Table 2 contains an overview of the total number of questions used for training and test purposes. We train on nearly 36,000 questions and test our pipeline on up to 1.8 million questions.

4.2 Predicting the Edit Type

Given a question which has been flagged as "to edit" in the first step, this processing step determines which aspect(s) of the question require an edit.

The 1000 annotated questions feature an average of 3.05 ± 1.84 edits. Three trusted annotators evaluated disjoint sets of 300 questions each. Additionally, a common set of 100 questions were labelled by all three annotators to test the agreement. The inter-annotator agreements for the four edit categories are shown in Table 3.

Edit Type	Code	SEC	Detail	Attempt
Kappa	0.67	0.59	0.19	0.65

Table 3: Inter-annotator agreement of edit category annotation, measured by Fleiss' Kappa.

The number of questions belonging to each category are reported in Figure 2. We used a majority consensus approach to determine the category of the 100 overlapping questions. Recall, that we annotate every edit of a question, and thus the total number of items shown in Figure 2 exceeds 1000. Of all edits, 30.75% could not be assigned to any of the four categories. We did not observe significant differences between the edit type distribution at different edit iterations (i.e. first edits are similarly distributed to second or third order edits).

We observe that *Code*, *SEC* and *Attempt* are often occurring categories, indeed more than half of the questions

	#Questions Overall	#Edited Questions	#Non-edited Questions
Test: Extreme	14,920	7,460	7,460
Test: Confident	85,072	42,536	42,536
Test: Ambiguous	1,772,649	522,874	1,249,775
Training: Extreme	35,892	17,946	17,946

Table 2: Basic statistics of our training and test data for the edit prediction task. Since more non-edited than edited questions exist, for the *Extreme* and *Confident* partitions, the number of non-edited questions was matched to the number of edited questions by sampling a subset of all questions in the respective dataset.

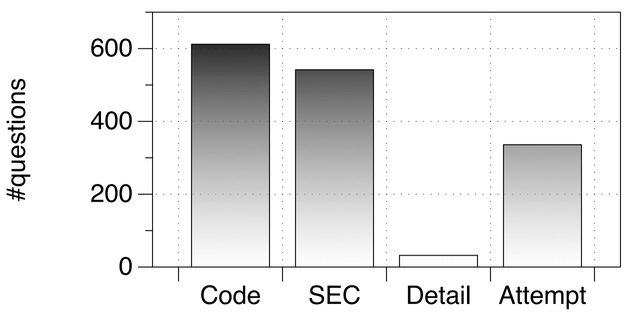

Figure 2: Annotation study results: number of questions with an edit from a particular category. The SEC category captures the problem Statement, Examples and the Context.

have at least one *Code* edit (it is also not uncommon to have several). For these three categories the inter-annotator agreement is also moderate to high (0.59 or higher). In contrast, the category *Detail* suffers both from very low inter-annotator agreement and few positive annotation results.

We train three binary classifiers, dropping the *Detail* category from further experiments due to the annotator disagreement and the small sample size. All questions with a particular edit type belong to the positive class for that edit type classifier, the remaining questions of our annotation set form the negative class. The classifier training follows a similar setup to step one. We derive features from the original question and include it in the training set for a classifier if at least one of the question's edit was annotated as belonging to the classifier's category. Due to the small size of the training data though we cannot rely on word unigrams as features. To avoid overfitting, we employ Latent Semantic Analysis [8] and rely on the 100 most significant dimensions as features. To evaluate the edit type prediction task, we use 5-fold cross validation.

5. EXPERIMENTS

We first present the results of our edit and edit type prediction tasks. Subsequently we present an analysis of a number of user-dependent factors that we hypothesise to influence the likelihood of a posted question requiring an edit (based on the hypotheses presented in Section 3.3).

5.1 Edit Prediction

The performance of our classifier on our test sets is presented in Table 4. As expected, the best results are achieved for the *Extreme* test set with an F1 score of 0.7. The recall of 0.78 implies that most questions which require an edit are classified as such by our approach, thus clearly demonstrating its feasibility. The classifier is trained on a feature set with a total of 7,206 features.

Test type	Precision	Recall	F1
Extreme	0.63	0.78	0.70
Confident	0.58	0.69	0.63
Ambiguous	0.51	0.65	0.57

Table 4: Classifier performance on the edit prediction task across our three test sets.

When comparing the performance of *Extreme* and *Ambiguous*, the impact of the test set generation process becomes evident. For the *Ambiguous* test set the performance of all three measures drops significantly. This is not surprising, as the middle ground questions (containing small edits or being poorly phrased but remaining unedited) are the most difficult for a classifier to identify correctly. We conclude that our proposed classifier, if employed on the stream of new Stack Overflow questions, would be able to spot the most severe cases of questions requiring an edit with high accuracy. We leave the exploitation of more advanced machine learning models and additional features for future work.

Important Features.

One of the benefits of a regression-based classifier is the ability to gain insights about the importance of different features based on the feature coefficients. In Table 5 we list the features (unigrams) with the highest and lowest coefficients respectively (after feature normalization). For instance, the term *microsoft* is an important feature for to-be-edited questions, while *lexer* is negatively associated with question edits, presumably because users discussing lexers have specific problems and a relatively deep understanding of their topic.

5.2 Edit Type Prediction

We now consider step 2 of our pipeline - the prediction of the type of edit(s) required to create a well-formed question. The results are shown in Table 6, rows one to three.

While the edits of *Code* and *SEC* can be predicted with moderate to high accuracy, the prediction of the *Attempt* category is essentially random.

Automatically Augmenting the Training Data.

Having so far relied on our manually annotated data only, we now turn to an automatic approach to augment the training data (the test data is fixed to our manually annotated

Strategy	Edit category	Nr. positive	Nr. negative	Precision	Recall	F1
No augmentation	Code	612	388	0.63	0.83	0.71
	SEC	542	458	0.57	0.62	**0.59**
	Attempt	336	664	0.39	0.45	0.40
Positive augmentation	Code	8157	338	0.63	<u>0.92</u>	<u>0.75</u>
	SEC	542	458	0.57	0.62	0.59
	Attempt	2387	664	<u>0.40</u>	<u>0.49</u>	<u>0.44</u>
Positive+ negative augmentation	Code	8157	8157	0.63	<u>0.95</u>	<u>**0.76**</u>
	SEC	542	542	0.55	0.49	0.52
	Attempt	2387	2369	0.38	<u>0.56</u>	<u>**0.45**</u>

Table 6: Classifier performance on the edit type prediction task. Numbers underlined are the ones higher than previous classification version. The best F1 scores in all edit type prediction tasks are highlighted in bold. Note that Nr. positive and Nr. negative only indicates the number of questions that affect training of the classifier. Precison, Recall and F1 are calculated based on the 1000 annotated questions.

Unigram	Coef.	Unigram	Coef.
dbcontext	0.88	mental	-0.29
microsoft	0.57	nicer	-0.31
xx	0.57	understood	-0.31
com	0.55	pre-compil	-0.34
tick	0.47	lexer	-0.41
neater	0.46	c/c++	-0.42
byte	0.45	firstnam	-0.47
inbuilt	0.44	testabl	-0.53
socket	0.42	string	-18.48
reproduc	0.39	archiv	-19.94

Table 5: Regression coefficients of the most positively and negatively weighted features (unigrams) for the edit predictiont ask.

questions). The goal is to provide sounder evidence on the performance of our predictors. We test two augmentation strategies:

1. **Positive augmentation**: we assume that questions with the term `code` appearing in the edited version while not in the original version have a big chance to be a positive question of edit type *Code*; this is verified in our annotated dataset where this is true for more than 38% of the questions in the edit type *Code* category. We use this strategy to collect additional training data from the *Extreme* training set; for the edit type *Code* we identified nearly 7000 additional questions. We followed the same approach for the *Attempt* category, relying on the term `tried` (this assumption holds true for 21% of our annotated data set). No augmentation was performed for category *SEC*, as no indicative terms could be determined.

2. **Negative augmentation**: We consider non-edited question in the *Extreme* training set as well-formed questions, and include similar number as edited questions to be the instances of the negative class.

To ensure that the classification results are not influenced by our selection criteria, the features `code` and `tried` are removed in the training phase.

The classifier performance with both types of enlarged training data are reported in Table 6, rows four to nine. In the case of positive augmentation it can be observed that both the *Code* and *Attempt* prediction performances increase. The improvements in F1 stem from an increase in recall. This is natural since the augmented training data contains only positive questions.

After negative questions were added as well, the edit type predictions *Code* and *Attempt* are very slightly enhanced. This indicates that the negative questions does not contain much information of each other. For type *SEC* the classifier performs as poorly as a random baseline.

To summarise, we have found that the edit prediction task can be solved with high accuracy, while the edit type prediction task is more difficult to solve. We have presented strategies to semi-automatically enlarge the training data which have been shown to be beneficial for the *Code* and *Attempt* categories.

5.3 Hypotheses Testing

We now turn to an analysis of our hypotheses presented in Section 3.3.

Up to now we have only considered the question content in edit and edit type prediction. We now explore the impact that different factors can have on the quality of a question. Such factors include the topic of a question, the user's prior experience on Stack Overflow, user knowledge on the question's topic, and the temporal influence of Stack Overflow. We first test our **hypotheses H1-H5**, then add related features for the prediction tasks to our classifier to investigate whether they can make a difference.

5.3.1 Topical Influence

We investigate **hypothesis H1**, i.e. if questions about particular frameworks or languages (e.g. `JavaScript`, `Java`), in particular those often used by programming beginners, are more prone to requiring an edit than questions related to more advanced topics such as software engineering (e.g. `design-patterns` or `compilers`).

For simplicity, we consider the tags assigned to each question as indicator of a question's topic. To avoid the influence of insignificant edits, we consider all questions of the *Confident* datasets (both training and test). Since a question may be assigned multiple tags, a question may appear in multiple tag sets. We rank the tags according to:

$$\frac{\#questions\ with\ substantial\ edits}{\#questions\ without\ an\ edit} \quad (2)$$

filtering out all those tags that appear too infrequently in the data set. We consider this ranking to provide us with an indication of a community's amount of beginners.

Rank	Tag	Ratio	#Questions in *Confident*
1	`asp.net-mvc-4`	6.16	505
2	`jsf`	6.02	615
3	`symfony2`	5.57	338
4	`r`	4.34	2,067
5	`opencv`	4.10	402
6	`matlab`	4.02	981
7	`core-data`	3.91	446
8	`angularjs`	3.67	288
9	`mod-rewrite`	3.52	297
10	`asp.net-mvc-3`	3.50	1,443
	....		
192	`vim`	0.52	746
193	`visual-studio-2008`	0.50	921
194	`web-applications`	0.49	774
195	`oop`	0.45	2,711
196	`database-design`	0.45	1,220
197	`unit-testing`	0.44	1,526
198	`logging`	0.44	624
199	`testing`	0.41	849
200	`design`	0.34	1,386
201	`svn`	0.27	1,186

Table 7: Overview of the topics (tags) which contain the most and least edited questions. All available data was used to generate the rank and ratios. The last column shows the number of questions in the *Confident* data set.

Table 7 provides an overview of the ten most and least edited topics (identified by their tags) in our data set. As hypothesised, the top-ranking topics are those more framework or language related, while low-ranking topics are more generic or advanced. For instance, `asp.net` questions usually require a lot of edits. In contrast, topics like `design` or `testing` require edits with a considerably lower likelihood.

We also report the number of questions a tag is assigned to in the *Confident* data set. It can be observed that the tags of most edited questions usually occur less than the non-edited ones (except the `r` tag). This indicates that not the large number of beginners leads to poorly phrased questions. It is more likely that these questions need to be edited because they are more complex and require more clarifications.

5.3.2 User Influence

Hypothesis H2 is concerned with the user effect - how does a user's familiarity with the portal Stack Overflow affect the probability of an edit? If **hypothesis H2** holds, we expect that the probability of a substantial edit decreases with increasing user experience with the platform. Such experience can be implied based on different types of user actions such as posting questions, answering, commenting or voting on postings.

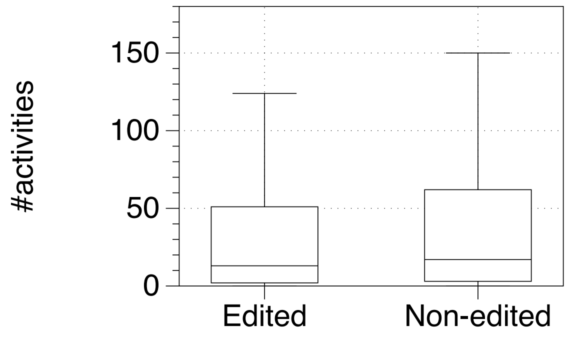

Figure 3: Influence of user experience on posting a question which requires an edit.

We use the *Confident* data set (training & test), which contains a total of 151,762 users – (16.4%) of all Stack Overflow askers. For each question, we determine the number of questions and answers in the entire data set (not limited to *Confident*) the asker has posted previously, then bin them into two groups: edited vs. non-edited questions. The comparison of these two groups is shown in Figure 3 in the form of a box plot. The number of past activities of a user is - as hypothesised - a significant indicator for the likelihood of a question edit. Users with fewer activities are more likely to edit their questions than more experienced users (to a statistical significant degree, p-value<0.001 by a Mann-Whitney test).

5.3.3 Knowledge Influence

Hypothesis H3 considers not only the activity of a user in the past (regardless of the topic), but also the knowledge of a user on a topic. In particular, we hypothesise that the number of questions requiring an edit decreases as a user gathers more experience on the topic (as she becomes more familiar with the terminology, etc.).

To evaluate this hypothesis, for each asker in the *Confident* data set (training+test) we plot the number of days since registering on Stack Overflow vs. the number of specific topic-related questions that require a substantial edit asked on this topic. As before, we use tags as topic indicator.

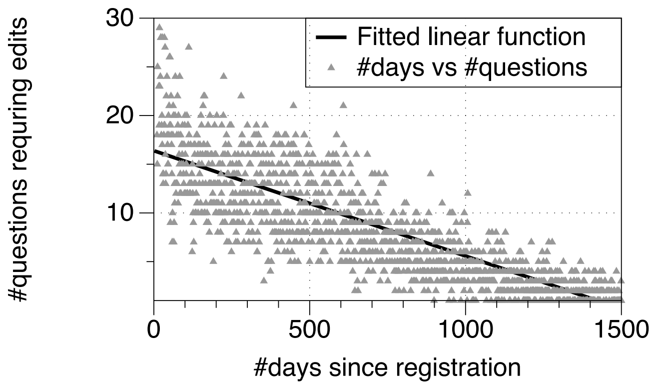

Figure 4: Influence of user knowledge on question edits. Results shown for topic (tag) `C#`.

Our analysis shows that these two variables are highly negatively correlated, with a Spearman correlation of -0.72 (p-value<0.001). We remove all users with a registration date older than 1500 days, and denote the activity of a user by a vector $(a_1, \ldots, a_{1500})$ where a_i denotes the number of questions and answers posted by this user at day i since his registration. Figure 4 shows the cumulative vector for all users involved in the topic `C#`. It can be observed that as time passes, a user asks less questions that require substantial edits. Though we only present the results for `C#`, we note that we observe the same trends for the top 20 topics (tags) on Stack Overflow, which include `Java`, `iOS` and `Python`.

5.3.4 Temporal Influence

Similarly to **hypotheses H2** and **H3**, we can also evaluate **H4** by considering all questions posted in a particular year. If **H4** holds, we expect to see a decreasing trend in questions requiring an edit. There is an influential factor, though, which will lead to more questions that require edits: new users registering and asking questions. Figure 5 plots:

$$E = \#\textit{edited questions} - \#\textit{non-edited questions}$$

in the *Confident* data partition over time, while Figure 6 depicts the evolution of user registrations in the same time period.

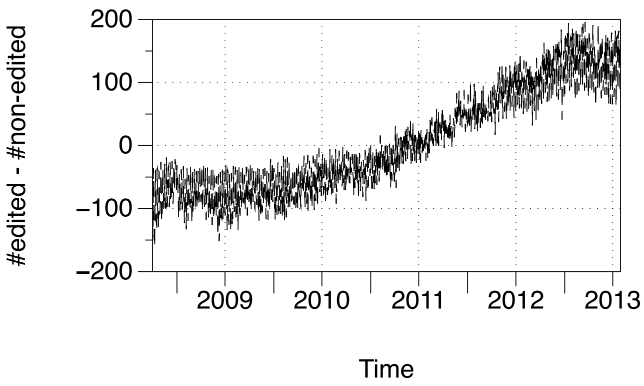

Figure 5: Overview of the gradual increase in edited questions on Stack Overflow over time.

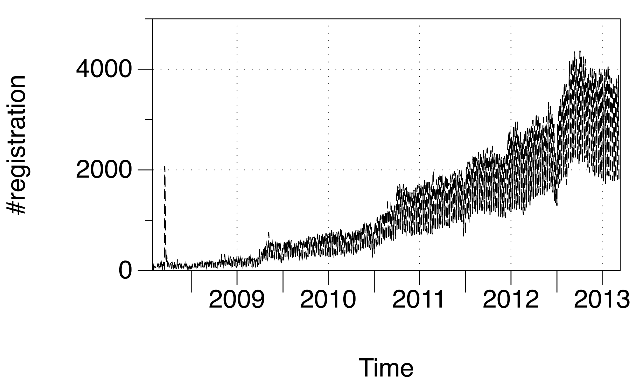

Figure 6: User registration over time.

The Spearman correlation between E and the number of user registrations is 0.79 with a p-value<0.001. This result provides additional support to the motivations of our work, as it shows that, despite the fact that an individual user asks fewer questions when he stays longer on Stack Overflow, the increasing popularity of the platform leads to the creation of several more questions that could benefit from a systematic assessment of their quality.

5.3.5 Influence of User "Age" on Edit Type

Hypothesis 5 is concerned with the role that user seniority plays in influencing the types of information (*Code*, *Attempt*, or *SEC*) that are (not) initially included in the questions.

For each of the 1000 annotated questions, we calculate the age of the question as the difference between its posting date and the registration date, in Stack Overflow, of its asker.

Figure 7 depicts the difference, in terms of age, of edited and non-edited questions in the context of the *Code* edit type: we observe that this type of edits is significantly (p-value<0.001 by a Mann-Whitney test) more likely to occur in the early days of a user's activity on the platform; *SEC* and *Attempt* edits do not show significant differences.

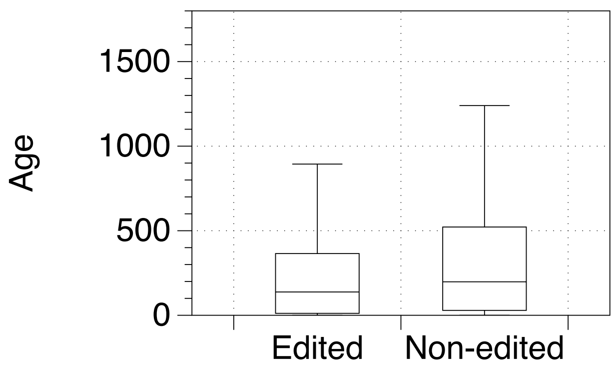

Figure 7: Influence of user age on posting a question which requires a *Code* type edit.

5.3.6 Influence on Prediction

In a final experiment, we created additional features for edit and edit type prediction based on the results of the investigated hypotheses. The following features were added to the existing feature set: 1) tags of a question, 2) #activities of the asker, 3) #days between the registration of the asker and the time she posted the question, and, 4) #days between a question was posted and the time Stack Overflow was launched.

In our experiments we did not observe substantial differences in F1 when adding those features to our original (unigram-based) feature set. This indicates that the content, i.e., the terms in a question, are more important that contextual factors for predicting the question (type) edit.

6. CONCLUSIONS

As cQA systems grow in popularity and adoption, the ability to provide automated quality enhancement tools is a key factor to guarantee usability, reliability, and high knowledge creation quality. In this paper we explored a specific aspect of user contributions: the formulation of well-formulated questions. In order to receive useful answers, a question should feature positive characteristics such as specificity (i.e. provide enough details to understand the nature of the problem), and clarity (i.e. provide examples, or personal experiences).

We analysed the editing behaviour of Stack Overflow users, and identified three main classes of useful editing actions. We then applied machine learning techniques to define an approach for the automatic suggestion of edit types for newly created questions. With respect to the research questions listed in Section 1 we can draw the following conclusions:

- **RQ1**: Question edits are a very good indicator of the quality of a given question, as their presence is also a reflection of several distinct traits of the asker (e.g. being new to a given technology, knowledge in the targeted topic, etc.).
- **RQ2**: Using a simple unigram model, we observe classification accuracies (F1) between 63% and 70%. This is a very promising result which indicates the possibility for significant improvements when adopting more sophisticated techniques.
- **RQ3**: Out of three identified classes of edits, only one (namely *code refinement*) features good prediction performance. The results are encouraging, but suggest that a more in-depth analysis of the different type of editing actions is required, to gain a better understanding of their features.

In addition to improvements to the components of our current question editing suggestion method, future work includes the extension of our analysis to other domains covered by the Stack Exchange platform (e.g. math, literature, etc.), to collect more insights about the editing behaviour of users across different knowledge domains.

Acknowledgements

This publication was supported by the Dutch national program COMMIT. This work was carried out on the Dutch national e-infrastructure with the support of SURF Foundation.

7. REFERENCES

[1] J. Ahn, B. S. Butler, C. Weng, and S. Webster. Learning to be a Better Q'er in Social Q&A Sites: Social Norms and Information Artifacts. *ASIST*, 50(1):1–10, 2013.

[2] A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec. Discovering Value from Community Activity on Focused Question Answering Sites: A Case Study of Stack Overflow. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 850–858, 2012.

[3] A. Anderson, D. Huttenlocher, J. Kleinberg, and J. Leskovec. Steering User Behavior with Badges. In *Proceedings of the 22Nd International Conference on World Wide Web*, WWW '13, pages 95–106, 2013.

[4] M. Asaduzzaman, A. S. Mashiyat, C. K. Roy, and K. A. Schneider. Answering Questions About Unanswered Questions of Stack Overflow. In *Proceedings of the 10th Working Conference on Mining Software Repositories*, MSR '13, pages 97–100, 2013.

[5] M. Bouguessa, B. Dumoulin, and S. Wang. Identifying Authoritative Actors in Question-answering Forums: The Case of Yahoo! Answers. In *Proceedings of the 14th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '08, pages 866–874, 2008.

[6] M. Burns and X. Kotval. Questions About Questions: Investigating How Knowledge Workers Ask and Answer Questions. *Bell Labs Technical Journal*, 17(4):43–61, 2013.

[7] D. Dearman and K. N. Truong. Why Users of Yahoo!: Answers Do Not Answer Questions. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '10, pages 329–332, 2010.

[8] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by Latent Semantic Analysis. *JASIS*, 41(6):391–407, 1990.

[9] B. V. Hanrahan, G. Convertino, and L. Nelson. Modeling Problem Difficulty and Expertise in Stackoverflow. In *Proceedings of the ACM 2012 Conference on Computer Supported Cooperative Work Companion*, CSCW '12, pages 91–94, 2012.

[10] F. M. Harper, D. Raban, S. Rafaeli, and J. A. Konstan. Predictors of Answer Quality in Online Q&A Sites. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '08, pages 865–874, 2008.

[11] T. Iba, K. Nemoto, B. Peters, and P. A. Gloor. Analyzing the Creative Editing Behavior of Wikipedia Editors: Through Dynamic Social Network Analysis. *Procedia - Social and Behavioral Sciences*, 2(4):6441 – 6456, 2010.

[12] P. Jurczyk and E. Agichtein. Discovering authorities in question answer communities by using link analysis. In *Proceedings of the sixteenth ACM Conference on Information and Knowledge Management*, CIKM '07, pages 919–922, 2007.

[13] J. Liu, Q. Wang, C.-Y. Lin, and H.-W. Hon. Question Difficulty Estimation in Community Question Answering Services. In *EMNLP*, pages 85–90, 2013.

[14] X. Liu, W. B. Croft, and M. Koll. Finding experts in community-based question-answering services. In *Proceedings of the 14th ACM International Conference on Information and Knowledge Management*, CIKM '05, pages 315–316, 2005.

[15] D. Ma, D. Schuler, T. Zimmermann, and J. Sillito. Expert recommendation with usage expertise. In *IEEE International Conference on Software Maintenance*, ICSM '09, pages 535–538, 2009.

[16] L. Mamykina, B. Manoim, M. Mittal, G. Hripcsak, and B. Hartmann. Design Lessons from the Fastest Q&a Site in the West. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pages 2857–2866, 2011.

[17] S. Nasehi, J. Sillito, F. Maurer, and C. Burns. What makes a good code example?: A study of programming Q&A in Stack Overflow. In *IEEE*

International Conference on Software Maintenance, ICSM '12, pages 25–34, 2012.

[18] A. Pal, S. Chang, and J. A. Konstan. Evolution of Experts in Question Answering Communities. In *Proceedings of the International AAAI Conference on Weblogs and Social Media*, ICWSM '12, pages 274–281, 2012.

[19] A. Pal, F. M. Harper, and J. A. Konstan. Exploring Question Selection Bias to Identify Experts and Potential Experts in Community Question Answering. *ACM Trans. Inf. Syst.*, 30(2):10:1–10:28, 2012.

[20] Y. R. Tausczik, A. Kittur, and R. E. Kraut. Collaborative Problem Solving: A Study of MathOverflow. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '14, pages 355–367, 2014.

[21] C. Treude, O. Barzilay, and M. Storey. How do programmers ask and answer questions on the web?: NIER track. In *Proceedings of the ACM/IEEE International Conference on Software Engineering*, ICSE '11, pages 804–807, 2011.

[22] B. Vasilescu, A. Serebrenik, P. Devanbu, and V. Filkov. How Social Q&A Sites Are Changing Knowledge Sharing in Open Source Software Communities. In *Proceedings of the 17th ACM Conference on Computer Supported Cooperative Work & Social Computing*, CSCW '14, pages 342–354, 2014.

[23] L. Yang, M. Qiu, S. Gottipati, F. Zhu, J. Jiang, H. Sun, and Z. Chen. CQArank: Jointly Model Topics and Expertise in Community Question Answering. In *Proceedings of the 22nd ACM International Conference on Conference on Information and Knowledge Management*, CIKM '13, pages 99–108, 2013.

[24] J. Zhang, M. S. Ackerman, and L. Adamic. Expertise Networks in Online Communities: Structure and Algorithms. In *Proceedings of the 16th International Conference on World Wide Web*, WWW '07, pages 221–230, 2007.

[25] Y. Zhou, G. Cong, B. Cui, C. S. Jensen, and J. Yao. Routing Questions to the Right Users in Online Communities. In *IEEE 25th International Conference*