A Hybrid Auto-tagging System for StackOverflow Forum Questions

Smrithi Rekha V
Center for Research in Advanced Technologies for Education
Amrita Vishwa Vidyapeetham
Kerala, India
smrithirekha@gmail.com

Divya N
Department of Computer Science and Engineering
Amrita Vishwa Vidyapeetham
Coimbatore, India
divyananjan261190@gmail.com

Bagavathi Sivakumar P
Department of Computer Science and Engineering
Amrita Vishwa Vidyapeetham
Coimbatore, India
pbsk@cb.amrita.edu

ABSTRACT
StackOverflow (SO) forum is a widely used platform for people to interact on topics related to Computer Programming languages. With more than three lakh users and ten lakh questions, StackOverflow is emerging as the biggest QA forum for programmers. The questions on StackOverflow cover a wide range of topics and are categorized using appropriate tags. Currently the tags are entered manually by users depending on their judgment of the tags. Since there are a huge number of tags, it is often a cumbersome process to search the correct tags. It may be useful to have an auto-tagging system that suggests tags to users depending on the content of the question.

In this paper we present a hybrid auto-tagging system for SO. The auto-tagging system includes a) programming language detection system b) SVM based question classification system. This system will suggest tags once a user enters a question.

Keywords
QA Forums, Stackoverflow, Machine Learning, SVM Classifier

1. INTRODUCTION
Online QA forums like Quora and StackOverflow (SO) are emerging as platforms that enable large users to interact on common topics of interest. SO is specifically for people discussing on computer programming languages like C, C++ and Python. Currently SO has more than three lakh users and ten lakh questions. SO follows a Question and Answer format and there are strict standards on the manner in which users can interact on the forum. SO forum is used for many purposes including:

- Getting solutions to programming problems
- Getting generic information on programming languages
- Learning new concepts in programming

Over a period of time SO has gained the reputation of being a reliable forum where users get quick responses to their questions and that too with high level of accuracy. This has been possible by the various measures taken to motivate more and more users to contribute in a meaningful way. Some of the ways of motivating users are:

- Providing provision to vote questions that are of good quality
- Providing badges to questions and answers for their quality
- Providing badges to users for their participation in the forum
- Providing badges for moderating the discussions

Such mechanisms have ensured the quality of discussions taking place on SO as well ensured that users will continue contributing to the forum.

Questions on the SO forum are of different types ranging from open-ended questions to specific programming syntax errors. Users are asked to post questions accompanied by relevant tags. Questions on the forum are categorized using tags. There are currently about 30,000 unique tags and currently "java" seems to be most popular tag, indicating that there are more number of questions related to java. A tag is a word or phrase that describes the topic of the question. Every question should have at least one tag, and can have up to five tags. Tags can be newly created by the user (if the user has reputation above 1500), or can be chosen from the list of tags available in the site. Tags help experts in finding the relevant questions that they can answer. Tags can also be used to find questions that are relevant or interesting to an user. Given this huge number of tags, it may be difficult for users to manually search appropriate tags while posting questions. Also, only users with good reputation can add new tags which in a way limit normal users from suggesting new tags. In this paper we propose a hybrid auto-tagging system which a) detects what programming language the question is related to b) what tags may be suitable for the content of the question. Such an auto-tagging system will reduce the manual effort involved in searching and attaching appropriate tags as well as increase the accuracy levels of tags on the forum.

Our paper is organized as follows. In Section 2 we have provided background information on SO forum and related work. Section 3 has a brief description of the recent survey that we conducted of SO users. In Section 4 we have provided details of implementation and discussed results in Section 5. We have concluded in Section 6.

ICONIAAC '14, October 10 - 11 2014, Amritapuri, India
Copyright 2014 ACM 978-1-4503-2908-8/14/08 $15.00.
http://dx.doi.org/10.1145/2660859.2660970

2. BACKGROUND AND RELATED WORK

SO forum data has been a pool for research from many perspectives. Researchers have analyzed user contributions, type of questions, benefit of the badges etc. These researchers have used several techniques like topic modeling, machine learning and analytics on the data. Such research has helped to understand the power of QA forums and how far these QA forums are serving as learning platforms.

In [7], Hanrahan et al have analyzed complex questions on SO and how they are handled by experts. They use a number of factors like the life span of a question and reputation score of experts to find the relationship between the question complexity and answers by experts. Seyed et al in [6] categorize the questions and also propose metrics, based on statistical analysis, to decide the quality of questions that contain code snippets. The authors in [3] adopt a multi-pronged approach to analyze SO data. They analyse users who post questions as well as those who answer questions. They also analyze the behavioral traits of active users. In addition to this they perform topic modeling on the questions. Miltiadis Allamanis and Charles Sutton in [14] categorize the questions after performing topic modeling. They show the frequency distribution of various topics through heat maps. Truede et al perform a detailed analysis of SO questions and answer several research questions including what types of questions are answered and how such forums contribute to software development body of knowledge [4]. In [9], the authors have correlated programming knowledge and age. Asaduzzaman et al have specifically focused on unanswered questions on SO and their characteristics [12]. Tag prediction has been done in [11] using Bayesian probabilistic model.

In our previous work [15], we had clustered the users of SO forum based on their contribution and participation. Users were classified into four categories namely naïve users, surpassing users, experts and outshiners. We had used X-Means and Expectation Maximization algorithms for this purpose. We found that most active users belong to the expert category but do not perform beyond that to reach the outshiners category.

Questions in SO carry appropriate tags for the purpose of effective classification of questions. The number of tags questions carry varies depending on the nature of the question and the knowledge of the user. The distribution of tags to posts in shown in Figure 1. We see that there are more questions with less number of tags.

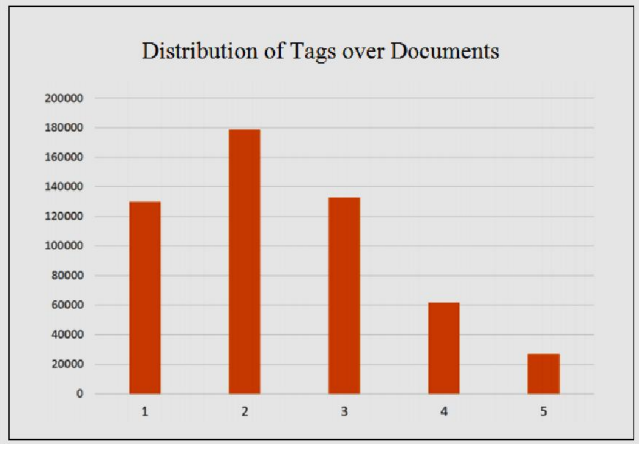

Figure 1: Tags-Posts Distribution

In this paper we use two main classifiers namely Multinomial Bayes and SVM. The Multinomial Naive Bayes Classifier is a probabilistic classifier which applies the Bayes Theorem with a naive independence assumption on the features extracted for text classification. Naïve Bayes classifier is a probabilistic classifier based on Bayesian Classification theorem which describes a supervised learning method associated with a statistical method for classification. It makes assumptions with a background probabilistic model and it helps us in capturing the uncertainty about the model by determining probabilities of the outcomes. The classifier assumes Class Conditional independence i.e the effect of the different variable values on a particular class are independent of each other. Such an assumption is made to make computation simple and hence it is named as "naïve" classifier [16]. The Bayes theorem applied to predict the probability that a given feature set belongs to a particular class label is given below:

$$P(labels, features) = P((labels) * P\ (features, labels)/P\ (features)$$

- *P (labels):* Is the probability of the occurrence of a particular label.
- *P (features, labels*): is the probability of a feature set being classified as that label.
- *P (features)*: is the prior probability of the occurrence of a feature set.
- *P (labels features*): This denotes the probability that the given feature(s) should have a particular label. A high value indicates high accuracy.

SVM is a popular machine algorithm which is mainly used for binary classification. SVM is a supervised learning model mainly used for analyzing and classifying data. Support vector machines are a well known set of methods for creating classifiers that can solve classification and regression problem. This is mainly used in case of binary classification where a given sample can be classified as belonging to one of the classes. In support vector machines the trained algorithm will build a model that can assign the new unseen data into any one of the class. In SVM model the examples are represented as points in the hyperspace mapping to their classes so that the example data of both the classes are separated by a wide gap. The new example data are also mapped to the same class and predicts the class it belongs to based on which side those examples are mapped.SVM can also perform non linear classifications efficiently using the kernel trick, where the mapping of inputs are done in feature space with high dimension. Therefore, SVM classifier supports in constructing models with single or set of hyperplanes in high dimensional space or even in infinite dimensional space mainly applied for solving classification and regression problems. A good number of positive examples and negative examples may be required to achieve accurate classification [13]. SVM has been applied to many real-world problems especially in the context of document and natural language processing.

3. USER PERCEPTION OF SO FORUM

To understand the impact of SO on users and the effectiveness of the SO forum, we had conducted a survey, the detailed analysis of which we propose to publish in our forthcoming work. We broadly discuss the results that are relevant to this paper. The survey was conducted online and we received 55 responses. Participants were contacted through

- Facebook groups

- Student groups on gmail
- Learning Analytics google group

Participants are programmers, students, researchers, Software Engineers and academicians. About 69% of the respondents are registered users of SO and hence can post, answer, vote and tag questions. About 45% of them have received badges on their questions and 88% have received votes on their answers. Respondents indicate that, predominantly they use the SO forum for programming error related questions and 42% of them find the forum useful for enhancing their coding capabilities.

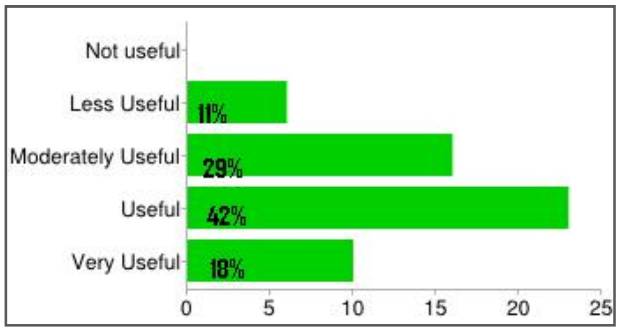

Figure: Usefulness of SO for enhancing programming capabilities

There were 2 questions which may be of relevance to this paper, which are:

- "Sufficiency of tags [On a scale of 1 to 5 rate your satisfaction levels with SO forum]"
- "Correctness of Tags [On a scale of 1 to 5 rate your satisfaction levels with SO forum]"

Only about 38% of them agree that the tags are sufficient and about 47% feel that the tags are only moderately accurate. Considering the user feedback on SO forum, we felt that we need to increase the sufficiency as well as correctness of the tags of posts. One of the ways of doing this is by automating the whole process so that tagging is not dependent on the user's understanding of the domain or pre-existing tags in the forum.

4. IMPLEMENTATION

SO forum allows users, who post questions, to enter upto five tags for a question. Entering all five tags makes a question more reliable and specific. While entering five tags is advantageous, most users find it easy to enter one or two tags but may not accurately enter other tags. This reduces the searchability of questions as well the granularity with which questions can be classified. Hence an auto-tagging system will be immensely beneficial. Auto-tagging of SO questions will have the following benefits:

- Assist the user in accurately tagging the questions
- Add more tags to existing questions
- Can assist new users of the forum to tag posts

We have developed an hybrid system to suggest tags for the posts in SO. The system comprises of a programming language detection system and linear SVM classifier.

The dataset is used from the corpus provided by Stackoverflow forum in the data explorer. There are about 50,000 questions in the Stackoverflow dataset. Each question consists of a title, body given in HTML format and the user defined tags. About 75% of dataset has been used as training data and the remaining for testing the model.

4.1 Programming Language Detection

Most questions are first tagged with the corresponding programming language. On observing the Stackoverflow posts, we found that most of them contain code snippets and tagged with the programming language names. Hence as the first step, our system will detect the programming language based on the code snippets. From the training data we collected the posts containing code snippets in their body, encapsulated within the specific html tags. We extract those snippets, tokenize them and also use regular expressions to find certain patterns matching in the programming code. These features are used to train a Multinomial Naive Bayes Classifier to identify the appropriate programming languages.

4.2 Question Content Based Tagging

The second system is the linear SVM classifier model to predict the tags based on the body content. We consider this problem as a binary classification problem to identify whether each tag t is suitable for the given post or not. The training data are constructed in such a way that for each tag-post pair the feature vectors are computed and used as positive training data examples. We need equal number of negative training examples to train our SVM model effectively. These negative training examples are constructed by taking some random tags and computing the feature vectors. Finally our SVM classifier is trained with all these training examples so that it can assign tags for the new posts. When a new post is given it traverses through all the tags and compute the feature vector for all tag-post pair to identify the suitability of a tag to a post. The steps for achieving this are as follows:

4.2.1 Pre-Processing

The posts are preprocessed in the following ways:

- The code snippets are extracted from the body content
- The title and the body of the post are tokenized and stemmed to obtain the meaningful terms.
- Regular expressions are used to strip frequent programming language tags like C++, python, java etc from the body and title
- The system checks if the tag is found in the title of the post
- The system checks if the tag is present in the body of the post
- It also checks if the token word is a hyphen separated word in the title (eg: visual-basic, python-image-library, machine-learning).
- A similar check is done for body of the post.
- The system loops through all the words in the title as well as the body to find sum over all the point wise mutual information values for the tag-word pair. PMI is defined as

$$PMI(t,w) = \lg \frac{P(t,w)}{P(t)P(w)}$$

The probability needed to calculate the PMI value is estimated by computing the maximum likelihood of the

training examples. The PMI value is set to 0 for the non observed post-tag pairs.

- Number of times the tag has occurred in the title and body is counted

The vector values of all the above features are used to train the SVM classifier.

4.2.2 Hybrid System

To build the hybrid system we combine the programming language detection system which produces the programming language tags on the observed code snippets and the SVM classifier producing N tags based on the content of the post. The outputs of both the systems are combined to generate tags for the new posts. The flow diagram of the hybrid system is shown in Figure 2.

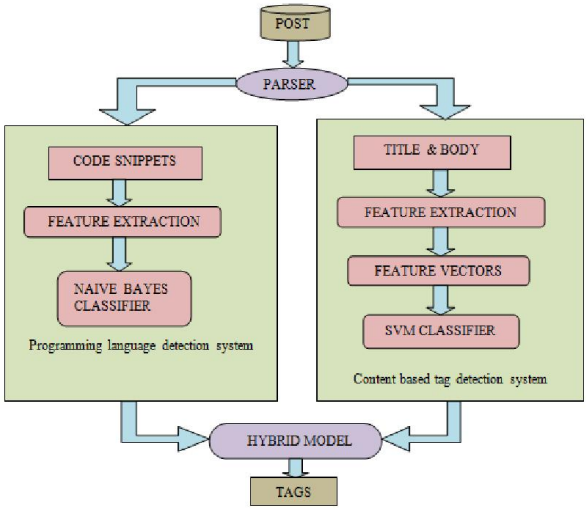

Figure 2: Flow Diagram of the Hybrid Classifier

Figure 3: Sample Question in the Tagger

5. RESULTS

Based on our hybrid classifier, we have obtained the following results based on our implementation:

We have been able to achieve 72% accuracy in tagging the questions. We get a precision value of .53 and recall value of 1 indicating that we were able to identify all of the relevant tags for a particular question and 50% of most relevant tags. In Figure 3 and Figure 4, we show a comparison between the tags created by our automated system and the actual tags provided by the user on the forum. The automated system shows high level of accuracy.

Figure 3: Tags generated for the question

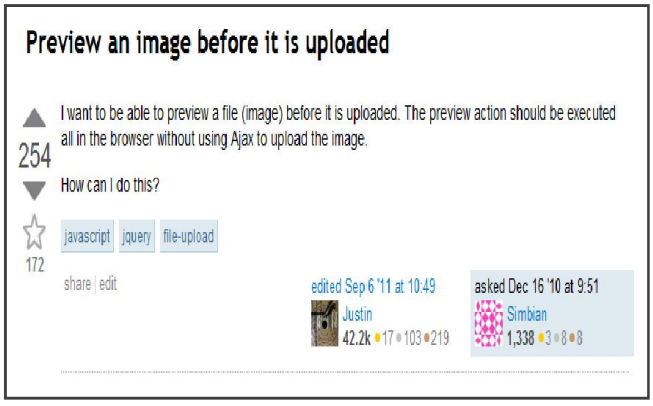

Figure 4: User generated on the SO website

6. CONCLUSION AND FUTURE WORK

In this paper we have discussed the implementation of a hybrid auto-tagging system for the posts of SO forum. We have also discussed the results obtained. Though the accuracy levels can be improved beyond 72%, this system will go a long way in assisting the user in accurately tagging the questions. As future work we propose to strengthen the algorithm to obtain more accuracy as

well analyze SO data from other perspectives. We also propose to conduct a survey of users to analyse the effectiveness of SO forum as a learning platform by understanding their perception of the quality of questions and answer, the accuracy of tags and the motivation levels to use the forum.

7. REFERENCES

[1] Saha, Avigit K., Ripon K. Saha, and Kevin A. Schneider. "A discriminative model approach for suggesting tags automatically for stack overflow questions." Proceedings of the 10th Working Conference on Mining Software Repositories. IEEE Press, 2013.

[2] Kim, Jinsuk, et al. "Automatic In-Text Keyword Tagging based on Information Retrieval." *JIPS* 5.3 (2009): 159-166.

[3] Wang, Shaowei, David Lo, and Lingxiao Jiang. "An empirical study on developer interactions in stackoverflow." *Proceedings of the 28th Annual ACM Symposium on Applied Computing*. ACM, 2013.

[4] Treude, Christoph, Ohad Barzilay, and Margaret-Anne Storey. "How do programmers ask and answer questions on the web?: Nier track." *Software Engineering (ICSE), 2011 33rd International Conference on*. IEEE, 2011.

[5] Correa, Denzil, and Ashish Sureka. "Fit or unfit: analysis and prediction of'closed questions' on stack overflow." *Proceedings of the first ACM conference on Online social networks*. ACM, 2013.

[6] Nasehi, Seyed Mehdi, et al. "What makes a good code example?: A study of programming Q&A in StackOverflow." *Software Maintenance (ICSM), 2012 28th IEEE International Conference on*. IEEE, 2012.

[7] Hanrahan, Benjamin V., Gregorio Convertino, and Les Nelson. "Modeling problem difficulty and expertise in stackoverflow." *Proceedings of the ACM 2012 conference on Computer Supported Cooperative Work Companion*. ACM, 2012.

[8] Kavaler, David, et al. "Using and asking: Apis used in the android market and asked about in stackoverflow." *Social Informatics*. Springer International Publishing, 2013. 405-418.

[9] Morrison, Patrick, and Emerson Murphy-Hill. "Is programming knowledge related to age? an exploration of stack overflow." *Mining Software Repositories (MSR), 2013 10th IEEE Working Conference on*. IEEE, 2013.

[10] Anderson, Ashton, et al. "Discovering value from community activity on focused question answering sites: a case study of stack overflow." *Proceedings of the 18th ACM SIGKDD international conference on Knowledge discovery and data mining*. ACM, 2012.

[11] Stanley, Clayton, and Michael D. Byrne. "Predicting Tags for StackOverflow Posts." *Proceedings of ICCM*. 2013.

[12] Asaduzzaman, Muhammad, et al. "Answering questions about unanswered questions of stack overflow." *Proceedings of the 10th Working Conference on Mining Software Repositories*. IEEE Press, 2013.

[13] Giménez, Jesús, and Lluis Marquez. "Fast and accurate part-of-speech tagging: The SVM approach revisited." *Recent Advances in Natural Language Processing III* (2004): 153-162.

[14] Miltiadis Allamanis, Charles Sutton,"Why, When, and What: Analyzing Stack Overflow Questions by Topic, Type, and Code.", MSR 2013, San Francisco, CA, USA

[15] Anusha S, Smrithi Rekha V, Bhagavathi Sivakumar P. "A Machine Learning Approach to Cluster the users of Stack Overflow Forum", ICAEES 2014, Tamilnadu, India

[16] Aggarwal, Charu C., and ChengXiang Zhai. "Mining text data". Springer, 2012.