UDK 004.852

LEARNING TO PREDICT CLOSED QUESTIONS ON STACK OVERFLOW

G. Lezina, A. Kuznetsov, P. Braslavski

Abstract

The paper deals with the problem of predicting whether the user's question will be closed by the moderator on Stack Overflow, a popular question answering service devoted to software programming. The task along with data and evaluation metrics was offered as an open machine learning competition on Kaggle platform. To solve this problem, we employed a wide range of classification features related to users, their interactions, and post content. Classification was carried out using several machine learning methods. According to the results of the experiment, the most important features are characteristics of the user and topical features of the question. The best results were obtained using Vowpal Wabbit – an implementation of online learning based on stochastic gradient descent. Our results are among the best ones in overall ranking, although they were obtained after the official competition was over.

Keywords: community question answering systems, large-scale classification, question classification.

Introduction

In recent years, Community Question Answering (CQA) services have become a good complement to the major web search engines and attracted a large audience. Users resort to the help of their peers on CQA platforms, when they want to get an answer to a complex information need that is hard to formulate as a short search query. On CQA sites users can describe the problem in detail, get a timely response from community members, clarify the issue if necessary, and leave feedback.

CQA services have also accumulated a huge amount of data. For example, Yahoo!Answers[1] claimed to have served one billion answers by May 2010[2]. Along with universal services like Yahoo!Answers, where users can ask questions on almost any subject, there are narrow-domain question&answer sites. Stack Overflow[3] is a service launched in 2008, where users ask and answer questions on software programming. At the time of writing (September 2013) Stack Overflow claims to have about 2 million registered users[4] and host more than 5.5 million questions[5]. Each question is provided with up to five tags that allow for classification and quick search over questions. The most popular tags are *c#, java, php, javascript, android, jquery, c++, python, html.*

The problem of content quality is crucial for web services relying on User-Generated Content (UGC). Purposely designed rules and incentives, feedback from community members, as well as dedicated moderators help to address this problem. Users on Stack Overflow can ask, answer, evaluate, and comment questions; score points and earn reputations. Users with high reputation receive additional privileges such as the ability

[1] http://answers.yahoo.com/
[2] http://yanswersblog.com/index.php/archives/2010/05/03/1-billion-answers-served/
[3] http://stackoverflow.com/
[4] http://stackoverflow.com/users
[5] http://stackoverflow.com/questions

to vote, flag posts as requiring moderator attention, leave comments and even edit posts owned by other users. Users with the highest reputation become moderators: they can lock posts (so the post cannot be voted, changed, etc.), suspend and ban users, approve tag synonyms, as well as close questions as inappropriate for Stack Overflow. The latter action is of primary interest of the study.

The question can be closed by a Stack Overflow moderator for one of the reasons: *off topic* (OT), *not constructive* (NC), *not a real question* (NRQ), and *too localized* (TL). Besides, there is an additional cause *exact duplicate* (ED), but this reason is beyond the scope of the study. Out of 6,000 questions posted on the service daily, about 6% end up closed by moderators.

Off topic (OT) questions fall out of the core Stack Overflow focus – software programming. Despite the following question closed as OT is related to programming documentation, it is not about programming *per se*:

Example: *There are plenty of extended training courses for Ruby on Rails, with devbootcamp, codestreak, etc. Are there any similar offerings for iOS development?*

Too localized (TL) question is unlikely to be helpful for anyone in the future; is relevant only to a small geographic area, a specific time point, or an extraordinary narrow situation that is not generally relevant to the global audience.

Example: *Is it time to start using HTML5? Someone has to start sometime but is now the time? Is it possible to use the new HTML5 tags and code in such a way as to degrade gracefully?*

Not constructive (NC) question does not fit well to Q&A format. While "good" questions imply facts, references, or specific expertise in answers, this sort of questions will likely solicit opinion, debate, arguments, polling, or extended discussion.

Example: *What is the best comment in source code you have ever encountered?*

Not a real question (NRQ) is an ill-formulated request. This type of questions is ambiguous, vague, incomplete, overly broad or rhetorical and cannot be reasonably answered in its current form.

Example: *Please let me know how to trace or debug the java script and jquery in visual studio 2008 and 2010. Because I have java script and jquery programme. Please let me know how to do it.*

The challenge organized by Stack Overflow on Kaggle platform offered the task of automatic prediction of closed questions along with the reason of closing[6]. The competition was held in August–November 2012 and attracted 167 teams as participants. In this paper, we consider the solution of this problem that was elaborated after the competition was over.

The paper is organized as follows. The next section surveys related work on CQA and Stack Overflow in particular, Sections 2 and 3 describe features used for question classification and employed machine learning techniques, respectively. Section 4 reports the classification results, considers them and defines the directions for future research.

1. Related Work

The phenomenon of the CQA services has become recently the subject of numerous studies. Content quality is the central issue of services based on UGC, CQA not being an exception.

[1] addressed the problem of automatic identification of high quality questions and answers in a dataset obtained from Yahoo!Answers. Using a wide range of features – content features, usage statistics, user relationships – the authors were able to separate

[6] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow/

high-quality items from the rest with a high accuracy. [2] introduced the question dichotomy *conversational* vs. *informational*, where the former questions are asked purely to start discussion and the latter are aimed at satisfying an actual information need. The authors implemented a binary classifier for these question types based on category, question text and asker's social network characteristics. Experiments were performed on data from three CQA services: Yahoo!Answers, Answerbag, and Metafilter. [3] investigated a similar facet of Q&A threads, namely *social* vs. *non-social* intent of the users: all questions intended for purely social engagement are considered social, while those that seek information or advice are considered non-social but instigating a knowledge sharing engagement. The dataset used in the study was comprised of 4,000 questions from two different CQA services. [4] introduces a notion of a high-quality question that supposes that the inquiry (1) attracts other users' attention; (2) fosters multiple answering attempts; (3) receives best answers within a short period of time. The authors analyzed factors affecting question quality and built a prediction algorithm. Experimental results performed on a dataset from the Entertainment & Music category of Yahoo!Answers demonstrated effectiveness of the approach.

A considerable number of studies has been done recently based on Stack Overflow data. An exhaustive list of scientific papers using Stack Overflow data can be found on a routinely updated page[7].

In the context of our work the following studies using Stack Overflow data are most relevant. [5] investigated the case of unanswered questions and revealed that 7.5% of all questions remained unanswered in Stack Overflow. Unanswered questions are related to closed questions since they often fall aside the main focus of Stack Overflow and are vaguely formulated. The authors performed a qualitative study of unanswered questions and subsequently tried to automatically predict how long a question will remain unanswered. [6] investigated topical structure and dynamics of Stack Overflow content. The authors applied Latent Dirichlet Allocation (LDA) and digested main topics of the forum: web-related discussions, database technology, development & deployment platforms, security, quality assurance, collaboration, knowledge/experience sharing, and general discussions. Extracted topics allowed for analyzing of trends, e.g. programming language, development platform and tools (such as version control systems) popularity over time. The paper [7] deals with closed questions on Stack Overflow and is the closest to ours. The paper provides a detailed overview of different features of closed questions: the life cycle of closed questions, the characteristics of users who ask these questions, the share dynamics of closed questions, the distribution of closed questions by type and the way the questions were closed (on moderator's own or based on users' voting), the topics of closed questions based on popular tags. The authors built a classifier for closed questions prediction using 18 features: user attributes (account age, badge score, previous posts with down votes), different scores gained by the asker, question content (# of URLs, # of popular tags), and text style features (title/body/code snippet length, # of punctuation marks, # of upper/lower case characters, etc.). We cannot compare the reported quality with our classification results, because the authors evaluated their method on a specially prepared balanced dataset.

2. Task Description

The main task was to build a classifier that assigns a question to one of the five classes: open question and four classes of closed questions mentioned earlier. In fact, participants were required to calculate five class probabilities for each question.

[7] http://meta.stackoverflow.com/questions/134495/academic-papers-using-stack-overflow-data

Table 1. Training dataset categories

Open	NRQ	NC	1.1cmOT	TL
3,575,678	38,622	20,897	20,865	8,910

2.1. Dataset. The organizers have made several datasets available for the participants. The main dataset comprises of all questions (3,664,927 questions in total) and answers presented on Stack Overflow since the service launch in 2008 until September 2012. The data contain question ID, its creation date, the age of the asker on the service, his/her reputation and the number of open questions at the time of posting, the question text, all answers, tags, and status (open or closed along with closing date in the latter case). The breakdown of the main dataset by classes of interest is presented in Table 1. As can be seen from the table, approximately 4% of all questions asked on Stack Overflow were closed for one of the four reasons; class sizes of closed questions are distributed very unevenly. Final evaluation of the classification results was carried out on data corresponding to two weeks of October 2012 (*private dataset*). Additionally, the participants were able to use a much richer data dump of Stack Overflow.

This dump contains additional information about 1.2 million registered users, posts information including history of the post editing, comments, etc.

The database dump structure is the following:

Badges. Badges is a user activity rewarding system on Stack Overflow.

Comments. This file contains information about who and when commented the post.

Post History. Contains information about posts' history and can describe one of the actions:

Initial, *Edit*, *Rollback Title/Body/Tags*. This information reflects all changes of the post's text.

Post Closed/Reopened/Deleted/Undeleted or Locked/Unlocked. This reflects information about all users and moderators' votes of the post.

Community Owned, *Post Migrated* (posts can be migrated through StackExchange sites if it is non-topic), *Question Merged/Unmerged*, *Question Protected/Unprotected*, *Post Disassociated* (an admin removes the OwnerUserId from a post).

Posts. Contains information about all questions and answers such as creation date, scores, best answer and so on.

Users. Contains user characteristics such as age, reputation, personal profile information and so on.

Votes. Contains information about all user votes – who voted and why. There are several reasons for voting:

AcceptedByOriginator (the answer to the question is accepted by the owner of the question), *UpMod/DownMod* (up or down votes for the question or answer), *Offensive* (the flag to eliminate offensive posts), *Favorite*, *Close/Reopen*, *BountyStart/BountyClose* (the users offer a reward for answering the question), *Deletion/Undeletion*, *Spam*, *InformModerator*.

The data section of the contest page contains direct links to data files as well as detailed description of file formats[8].

[8] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow/data

2.2. Evaluation Metrics. The results of the competition were evaluated using multiclass logarithmic loss (MLL) metrics calculated by the following formula:

$$\mathrm{MLL} = -\frac{1}{N}\sum_{i=1}^{N}\sum_{j=1}^{M} y_{i,j}\ln(p_{i,j}),$$

where N is the number of observations, M is the number of class labels, $y_{i,j}$ is 1 if observation i is in class j and 0 otherwise, and $p_{i,j}$ is the predicted probability that observation i is in class j.

To avoid infinite values all zero probabilities are rounded to a small value. The submitted probabilities are replaced with $\max(\min(p, 1-10^{-15}), 10^{-15})$.

2.3. Baselines. The contest organized provided three baselines: *uniform baseline* (all classes are equiprobable for all questions); *prior baseline* (all questions have equal probability distribution proportional to class sizes in training set), and *basic baseline*. The *basic baseline* is built using Random Forest classifier and incorporates six features:

OwnerUndeletedAnswersAtPostCreation – the number of answer posts that have been made by the time of asking.

BodyLength – initial post body length in characters (including code blocks).

ReputationAtPostCreation – user reputation at post creation time.

NumTags – number of tags assigned to the post (max 5).

TitleLength – post title length in characters.

UserAge – the time elapsed from the time the user registered with Stack Overflow.

Uniform, *prior*, and *basic* baselines deliver on private dataset 1.6, 0.47, and 0.46 MLL points, respectively.

3. Classification Features

Our work deals mostly with feature engineering aimed at a richer description of posts to be classified. Classification features we use can be grouped as follows: 1) user features; 2) user interaction features (calculated partially based on the Stack Overflow database dump); 3) shallow post features; 4) word unigrams, bigrams, and trigrams; 5) single tag and tag pair features; 6) LDA topic probabilities.

3.1. User Features (UF). User features describe the asker regardless of his/her question to be classified.

Reputation. User reputation by the time of creation of the Stack Overflow database dump. The hypothesis is that users with a high reputation ask better questions.

AgeFilled, *AboutMeFilled*, *LocationFilled*, *WebsiteFilled*, *AllInfoFilled*. This group of features reflects the completness of the user profile on Stack Overflow. The assumption is that a complete profile corresponds to a more "serious" user concerned with his/her online appearance.

UpVotes, *DownVotes*. Votes the user received for his/her posts.

3.2. User Interaction Features (UIF). This set of features is calculated based on the graph induced from the Stack Overflow database dump. Users are vertices in such a graph; there are three types of directed edges: 1) user A answered B's question; 2) user A voted up or down for B's question; 3) user A voted for closing B's question.

CVInDegree, *CVOutDegree*. Close votes received by the user's posts, close votes for other users' posts by the user.

QAInDegree, *QAOutDegree*. The total number of answers received/given by the user.

QAClustCoef. The local clustering coefficient of the users' question-answer network. The clustering coefficient reflects how close the immediate neighborhood of a vertex is to a complete graph, i.e. the density of QA interactions in the user's immediate proximity. The hypothesis is that users with high QA clustering coefficient are more communicable and tend to ask conversational questions.

3.3. Shallow Post Features (PF). All posts in the dataset were preprocessed as follows:

- Stack Overflow markup removed. Stack Overflow markup is used to allocate chunks of code, links and some other elements.
- Stopwords removed.
- Rare words removed. Most of the rare words are typos or together words. So we removed words that are used less than 100 times in the training dataset.
- Post and title text was stemmed using Porter stemmer implementation from NLTK[9] library.

Post features are calculated based on post title and body.

CBCount, LinkCount. Number of code blocks, links in the post body.

Dates, *Times.* Number of date and time mentions in the post body. Using this feature we tried to capture time-specific questions. As described in Stack Overflow FAQ, the *too localized* posts are closed as time or place specific.

NumberOfDigits. Number of digits in the post body.

NumberOfSentences. Number of sentences in the post body excluding code blocks.

NumberOfSentencesStartWithI. Number of sentences in the post body that start with "I". A high portion of sentences starting with the first person pronoun such as "I need your help. I do this... I get this... I need to get this..." can indicate a detailed though probably subjective problem description. It is more likely to be a *too localized* question than a *not constructive* one.

NumberOfSentencesStartWithYou. Number of sentences in the post body that start with "you". A high proportion of sentences of the kind can indicate an opinionated or conversational question.

UpperCaseLowerCaseRatio. Characterizes typographic neatness of the post.

FirstTextLineLength. The first short line in the post body implies usually a personal appeal or greeting.

NumberOfInterrogativeWords. Number of interrogative words – *what*, *where*, *why*, etc.

NumberOfSentencesStartWithInterrWords. Equals roughly to the number of interrogative sentences in the post.

Additionally, we used easily computable features such as punctuation marks count, indentation in code blocks, etc.

Tag features include single tag frequencies for each class and close frequencies for each tag pair. Our hypothesis was that some tag pairs can indicate disputable and controversial questions.

3.4. Content Representation (CF). We employed three approaches to content representation:

A. Single tag and tag pair frequencies for each class;

B. Unigrams, bigrams, and trigrams (NGr);

C. Topic modeling using Latent Dirichlet Allocation (LDA).

[9] http://nltk.org/

Table 2. *TL* most probable topics

Words	# of docs
java, eclips, jar, run, maven	27
java, org, spring, apach, hibern	25
screen, posit, top, left, bar	22
view, iphon, io, xcode, develop	21
rubi, rail, ruby-on-rail, rout, gem	19
android, activ, app, layout, emul	19

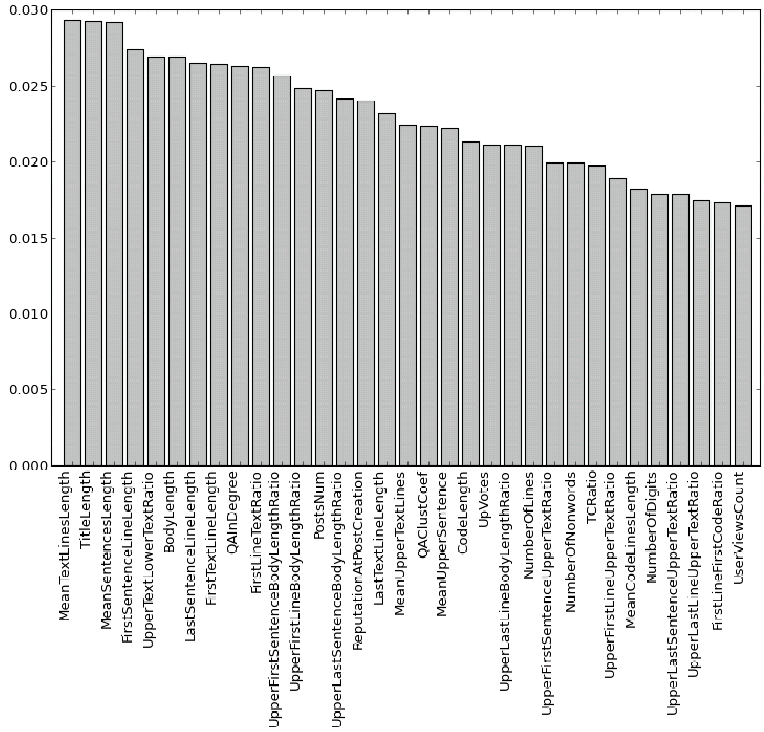

Fig. 1. *TL* topics distribution

While the n-grams approach allows one to capture collocations and technical terms, LDA produces a higher-level topical representation based on word co-occurrence in documents.

LDA [8] is an unsupervised technique that makes it possible to uncover topical structure of a document collection. We employed GibbsLDA++[10] implementation that uses Gibbs Sampling for parameter estimation and inference and built-in Vowpal Wabbit LDA implementation. The model was trained for 200 topics with 1,000 iterations.

Figures 1–4 show topic distributions for each class in the training dataset.

As we can see from Fig. 1 and Table 2, the most frequent topics of questions closed as *too localized* are related to Java programming. As expected, most frequent topics in *not constructive* class seem to relate to general questions, recommendations, and advice (see Fig. 2).

[10] http://gibbslda.sourceforge.net/

Table 3. *NC* most probable topics

Words	# of docs
develop, experi, softwar, peopl, compani	377
book, good, learn, tutori, recommend	194
program, languag, learn, student, cours	69
design, pattern, logic, ui, architectur	51
edit, support, featur, editor, use	48
version, git, repositori, svn, release	34

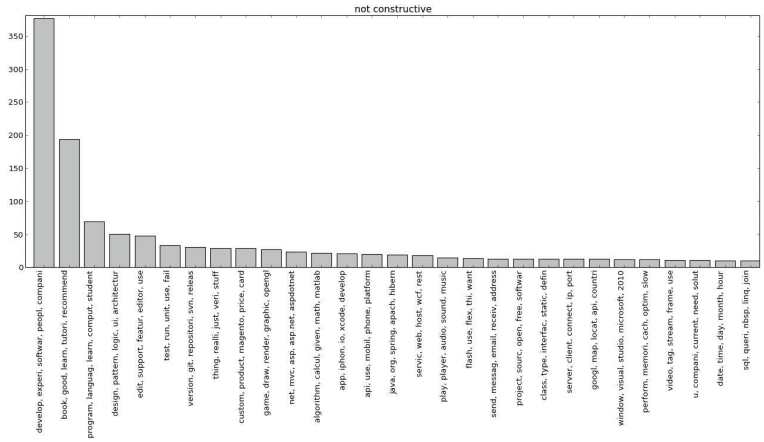

Fig. 2. *NC* topics distribution

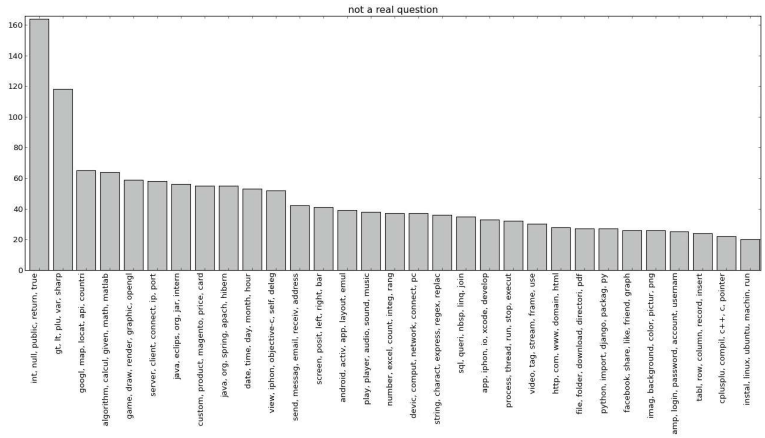

Fig. 3. *NRQ* topics distribution

3.5. Feature Importance. To select most important features we built a large number of trees for randomly selected subsets of features as described in [9]. The result was used while classifying data using Random Forest and Support Vector Machine classifiers. Figures 5–8 show relative importance of features described in Section 3 for *too localized*, *not constructive*, *not a real question* and *off topic* classes, respectively.

The sets of important features for *TL*, *NC* and *OT* are quite similar – they contain mostly textual features described in Section 3.3. It is interesting to note that for *NRQ* class the most important features are user reputation at post creation, user post count,

Table 4. *NRQ* most probable topics

Words	# of docs
int, null, public, return, true	164
gt, lt, plu, var, sharp	118
googl, map, locat, api, countri	65
algorithm, calcul, given, math, matlab	64
game, draw, render, graphic, opengl	59

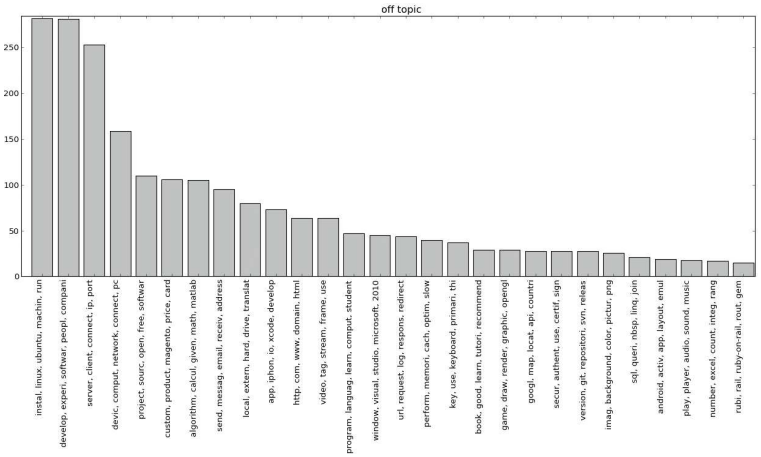

Fig. 4. *OT* topics distribution

Table 5. *OT* most probable topics

Words	# of docs
instal, linux, ubuntu, machin, run	282
develop, experi, softwar, peopl, compani	281
server, client, connect, ip, port	253
devic, comput, network, connect, pc	159
project, sourc, open, free, softwar	110

up votes that user received for his/her questions and/or answers and also number of close votes that the user received in the past.

4. Classification Methods

We experimented with three different machine learning methods: Random Forest (RF), Support Vector Machine (SVM) and Vowpal Wabbit (VW), an online learning implementation of stochastic gradient descent algorithm.

4.1. Random Forest. Random Forest is an ensemble approach. The main idea is that a set of "weak" classifiers form a "strong" classifier. In Random Forest each "weak" learner is a decision tree which takes an input into the top, passes it through so that input data are splitted into smaller sets chosen at random. Output is an average over all terminal nodes that are reached in each tree.

Random Forest was trained with the following parameters:

$n\_estimators = 50$ – the number of trees in the forest. More estimators mean

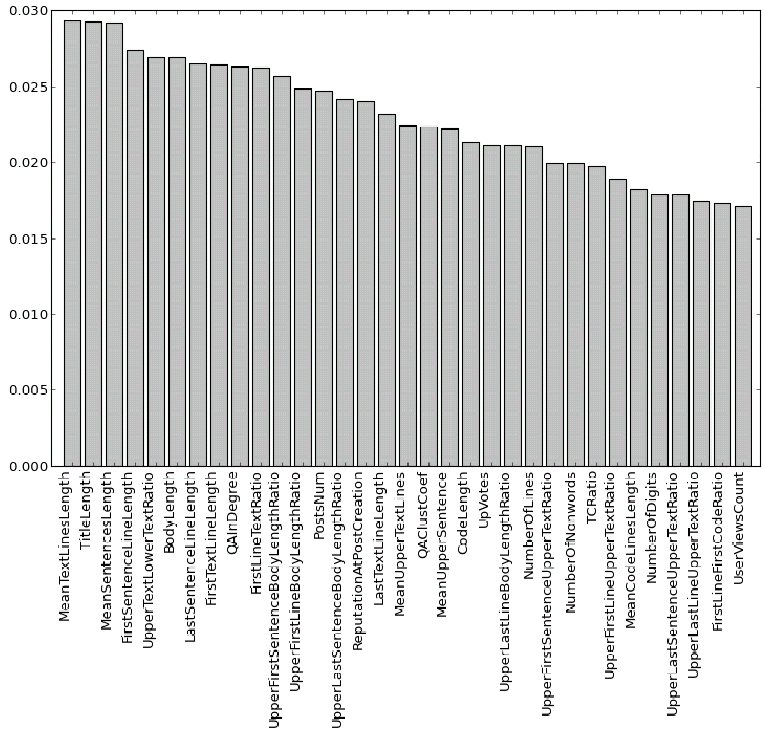

Fig. 5. Relative *TL* feature importance

in general better results but training time increases rapidly.

$min\_samples\_split = 2$ – the minimum number of samples required to split an internal node.

$min\_samples\_leaf = 1$ – the minimum number of samples in newly created leaves. The split is discarded if after the split, one of the leaves would contain less than specified value.

$max\_features = \sqrt{n\_features}$ – the number of features to consider when looking for the best split. Here $n\_features$ is the total number of features.

We have used Gini criterion for measuring best split when choosing a variable at each step in the decision tree construction process.

The maximum tree depth parameter was not specified explicitly so when building the tree the nodes were expanded until all leaves contained $min\_samples\_split$.

Bootstraped samples were used for building trees.

4.2. Support Vector Machine. Support Vector Machines (SVM) have been shown to be highly effective for traditional text categorization [10].

The basic idea of this approach is to find a hyperplane (or a set of hyperplanes) that is represented by vector $\vec{w}$ in a high-dimensional space which separates one class from another (in the case of binary classification problem). This separation (margin) should be as large as possible from the nearest training data points of any class. This is the optimization problem

$$\vec{w} = \sum a_j c_j \vec{s}_j,$$

where c_j is one of two classes label and $c_j \in \{-1, 1\}$, s_j is a sample to be classified and a_j is obtained by solving a dual optimization problem. The s_j for which $a_j >= 0$

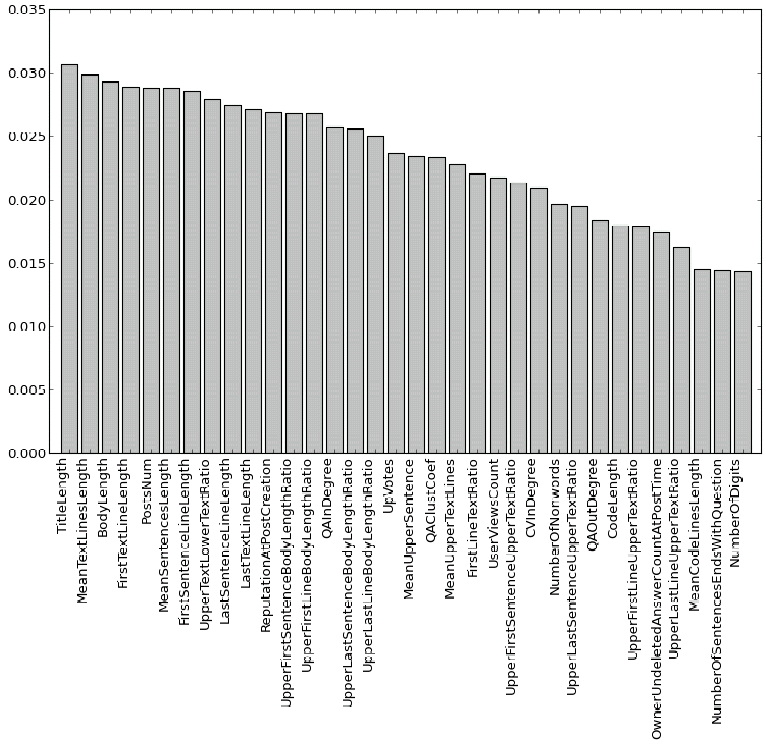

Fig. 6. Relative *NC* feature importance

is called a support vector.

We used liblinear[11] implementation of SVM. It was chosen because of amount of data. As mentioned above it is slightly less than 4 million of samples and we did not balance data as we did for RF classifier. Liblinear does not use kernels and can be trained very fast.

The parameters that we used in our experiments for liblinear algorithm implementation are the following:

penalty = $l1$ – the norm used in the penalization. In the case of $l1$ norm, the vectors of the weights assigned to the features are sparse.

loss = $l2$ – the loss function. Here $l2$ means the squared hinge loss (possible $l1$ value means hinge loss function).

tol = 0.001 – the tolerance for stopping criteria.

$C = 1.0$ – the penalty parameter of the error term. To estimate this parameter we applied Grid Search algorithm. This algorithm is a standard way for searching optimal hyperparameters.

In our case the number of samples is much more than the number of features so the algorithm was set to solve the primal problem with dual parameter.

As our classification task is a multi-class task we used one-vs-rest (OVR) strategy. In this case for the n classes n binary classifiers are trained.

The class_weight parameter was not defined explicitly so the values were adjusted inversely proportional to class frequencies.

[11] www.csie.ntu.edu.tw/c̃jlin/liblinear

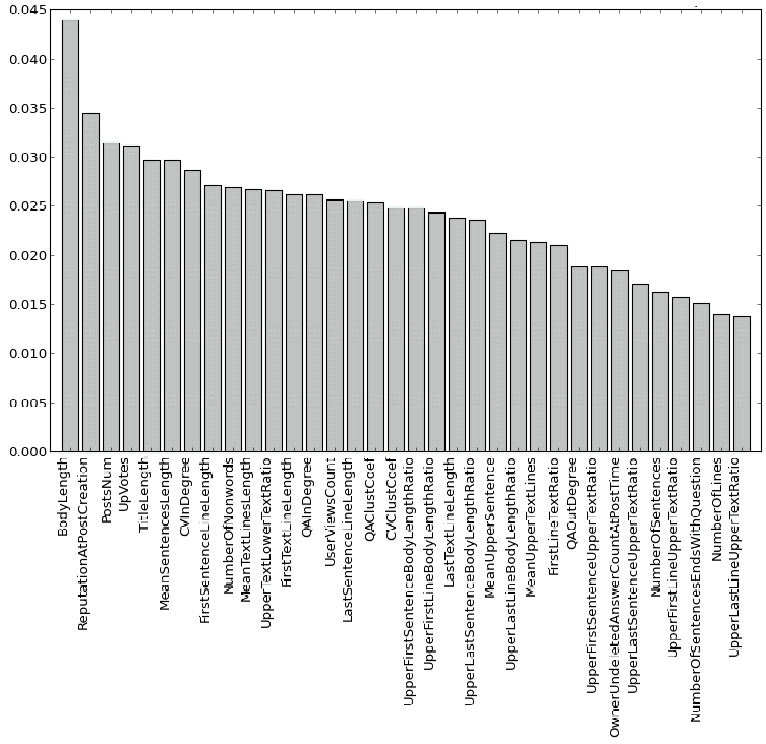

Fig. 7. Relative *NRQ* feature importance

4.3. Vowpal Wabbit. Vowpal Wabbit (VW)[12] is a library developed by John Langford. VW focuses on the approach to feed the examples to an online-learning algorithm [11] in contrast to parallelization of a batch learning algorithm over many machines. The default learning algorithm is a variant of online gradient descent.

When feeding training samples to VW in chronological order we got 0.315 MLL points on private dataset. Randomly shuffled data delivered 0.334 MLL points. The online algorithm is very sensitive to the last changes in the dataset. So the questions in the last year on Stack Overflow were closed more frequently (for each close reason) than in the first year.

Unlike the typical online learning algorithms which have at least one weight for every feature, the approach used in VW makes it possible to induce sparsity in learned feature weights. The main idea of truncate gradient is that it uses the simple rounding rule of weight to achieve the sparsity. The most of the methods rounds small coefficients by threshold to zero after a specified number of steps. The truncated online gradient descend algorithm described very well in [11].

We tuned some parameters for this algorithm which we found empirically to be the best:

logistic loss – optimization function,
$p = 0.5$ – power of learning rate decay,
$l = 1$ – learning rate,
$d = 1$ – decay learning rate.

[12] https://github.com/JohnLangford/vowpal_wabbit/

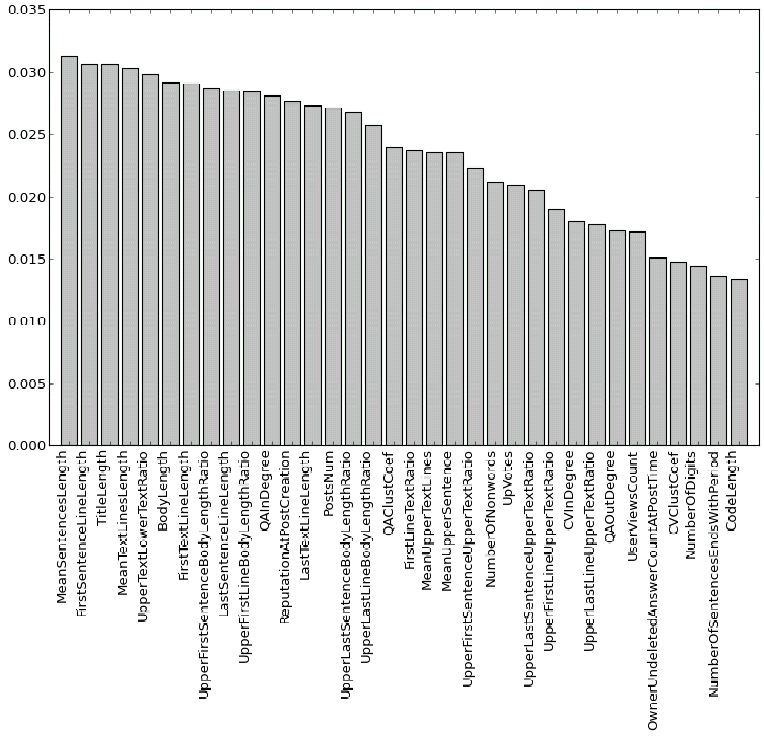

Fig. 8. Relative OT feature importance

We give a short explanation for these parameters. The form of the standard stochastic gradient descent (SGD) rule is

$$f(w_i) = w_i - \eta \nabla_1 L(w_i, z_i),$$

where η is the step size and here it is decaying and becomes smaller when i increases. Its value is updated according to:

$$\eta_e = \frac{l d^{n-1} i^p}{\left(i + \sum_{e'<e} i_{e'}\right)^p},$$

where l is the learning rate, d is the decay learning rate, i is the initial time for learning rate, p is the power of learning rate decay.

VW has built-in LDA which we used for 200 topics.

5. Classification Results

Prediction results were evaluated on public and private datasets provided by Kaggle[13] using MLL metrics. Class labels for public and private data are hidden and still not accessible for participants. So the result for this data can be calculated only on the Kaggle server.

Table 6 presents results for different methods we used; the best result of the official competition on Kaggle is also provided.

[13] https://www.kaggle.com/c/predict-closed-questions-on-stack-overflow/data

Table 6. MLL scores

Method	RF	VW	SVM
UF+PF+LDA	0.417	**0.315**	0.374
UIF+UF+PF+LDA	0.448	0.318	0.414
UIF+UF+PF+NGr	0.440	0.341	0.382
LDA	0.648	0.456	0.487
uniform baseline	1.609		
prior baseline	0.469		
basic baseline	0.461		
best result	0.298		

As we can see from Table 6 in VW outcome user interaction features worsen the results for a small value. The table demonstrates that the text features contribute much more to the result than the user features, whereas LDA outperforms n-grams approach.

Vowpal Wabbit, which gave us the best result. utilizes a logistic loss function and one against all classification. SVM and RF classifiers require preliminary LDA model construction that can be quite resource-consuming, while VW has online LDA implementation that does not require much time and memory.

As we mentioned earlier the baseline does not take into account post content and as we got convinced the text features are very informative.

6. Conclusion and Future Work

After some manual analysis we noted that some questions' status is open but it actually should be closed. Sometimes it is reflected in the comments and it would be useful to consider such recommendations that are available in the Stack Overflow database dump. It is a good idea to make a better use of the database dump not only for further text feature extraction but for observing user communication in this way. Some posts can be edited by different users no matter whether the posts are community or single user owned, so changes that make a "bad" post with votes for closing into a "good" one can be tracked.

Sometimes it is very hard to determine *too localized* question because it can be seen from the text of the post, although sometimes it is enough to look at the code included in the post body. We did not analyze the content of the code in any way during the classification. It is a nontrivial task to determine if the code works for specific conditions and will never be useful for anyone else in the future. It might be helpful to analyze stack traces that often appear in code blocks and are a good signal of too specific questions.

As we can see from the results, the user interaction features do not improve the results. For example it is said that for *not constructive* category, users tend to engage in discussion but it does not mean that such posts contain many comments or debates. We cannot match text patterns, which cause debates because this is not reflected in user behavior. We can only compare text patterns with the fact that the question was closed as not constructive. This set of features can bring some error in the result score. Text features work well as we can see from examples in Section 3.4, where some topics are typical for a given close reason.

This work is partially supported by the Russian Foundation for Basic Research, project No 14-07-00589 "Data Analysis and User Modelling in Narrow-Domain Social Media".

Резюме

Г. Лёзина, А. Кузнецов, П. Браславский. Прогнозирование закрытых вопросов на Stack Overflow.

В статье рассматривается задача прогнозирования вероятности того, что вопрос на сервисе Stack Overflow – популярном вопросно-ответном ресурсе, посвященном разработке программного обеспечения – будет закрыт модератором. Задача, данные и метрика оценки качества были предложены в рамках открытого конкурса по машинному обучению на сервисе Kaggle. В процессе решения задачи мы использовали широкий набор признаков для классификации, в том числе признаки, описывающие личные характеристики пользователя, взаимодействие пользователей друг с другом, а также содержание вопросов, в том числе тематическое. В процессе классификации протестировано несколько алгоритмов машинного обучения. По результатам эксперимента были выявлены наиболее важные признаки: личные характеристики пользователя и тематические признаки вопроса. Наилучшие результаты были получены с помощью алгоритма, реализованного в библиотеке Vowpal Wabbit, – интерактивного обучения на основе стохастического градиентного спуска. Наилучшая полученная нами оценка попадает в топ-5 лучших результатов в финальной таблице, но получена после даты завершения конкурса.

Ключевые слова: социальные вопросно-ответные системы, классификация большого объема данных, классификация вопросов.

References

1. *Agichtein E., Castillo C., Donato D., Gionis A., Mishne G.* Finding high-quality content in social media // Proc. 2008 Int. Conf. on Web Search and Data Mining. – N. Y.: ACM, 2008. – P. 183–194.
2. *Harper F.M., Daniel M., Konstan J.A.* Facts or friends?: distinguishing informational and conversational questions in social Q&A sites // Proc. 27th Int. Conf. on Human Factors in Computing Systems. – N. Y.: ACM, 2009. – P. 759–768.
3. *Rodrigues E.M., Milic-Frayling N.* Socializing or knowledge sharing?: characterizing social intent in community question answering // Proc. 18th ACM Conf. on Information and Knowledge Management. – N. Y.: ACM, 2009. – P. 1127–1136.
4. *Li B., Tan J., Lyu M.R., King I., Mak B.* Analyzing and predicting question quality in community question answering services // Proc. 21st Int. Conf. Companion on World Wide Web. – N. Y.: ACM, 2012. – P. 775–782.
5. *Asaduzzaman M., Mashiyat A.S., Roy C.K., Schneider K.A.* Answering questions about unanswered questions of stack overflow // Proc. Tenth Int. Workshop on Mining Software Repositories. – IEEE Press, 2013. – P. 97–100.
6. *Barua A., Thomas S.W., Hassan A.E.* What are developers talking about? An analysis of topics and trends in Stack Overflow // Empir. Software Eng. – 2014. – V. 19, No 3. – P. 619–654.
7. *Correa D., Sureka A.* Fit or unfit: Analysis and prediction of “closed questions” on stack overflow. – 2013. – arXiv:1307.7291.
8. *Blei D.M., Ng A.Y., Jordan M.I.* Latent dirichlet allocation // J. Mach. Learn. Res. – 2003. – V. 3. – P. 993–1022.
9. *Draminski M., Rada-Iglesias A., Enroth S., Wadelius C., Koronacki J., Komorowski J.* Monte Carlo feature selection for supervised classification // Bionformatics. – 2008. – V. 24, No 4. – P. 110–117.
10. *Joachims T.* Text Categorization with Support Vector Machines: Learning with Many Relevant Features // Proc. 10th Eur. Conf. on Machine Learning. – London: Springer-Verlag, 1998. – P. 137–142.

11. *Langford J., Li L., Zhang T.* Sparse online learning via truncated gradient // J. Mach. Learn. Res. – 2009. – V. 10. – P. 777–801.

Поступила в редакцию
10.09.13

Lezina, Galina – PhD Student, Ural Federal University; Software Engineer, SKB Kontur, Ekaterinburg, Russia.

Лёзина Галина – аспирант ИМКН, Уральский федеральный университет; инженер-программист, ЗАО «ПФ «СКБ Контур», г. Екатеринбург, Россия.

E-mail: *galina.lezina@gmail.com*

Kuznetsov, Artem – Master Student, Ural Federal University; Software Engineer, SKB Kontur, Ekaterinburg, Russia.

Кузнецов Артем – магистрант ИМКН, Уральский федеральный университет; инженер-программист, ЗАО «ПФ «СКБ Контур», г. Екатеринбург, Россия.

E-mail: *artem.m.kuznetsov@gmail.com*

Braslavski, Pavel – Senior Researcher, Ural Federal University; Head of Kontur Labs, SKB Kontur, Ekaterinburg, Russia.

Браславский Павел – кандидат технических наук, старший научный сотрудник, Уральский федеральный университет; руководитель отдела исследований и новых разработок, ЗАО «ПФ «СКБ Контур», г. Екатеринбург, Россия.

E-mail: *pb@kontur.ru*