Will My Question Be Answered? Predicting "Question Answerability" in Community Question-Answering Sites

Gideon Dror, Yoelle Maarek, and Idan Szpektor

Yahoo! Labs, MATAM, Haifa 31905, Israel
{gideondr,yoelle,idan}@yahoo-inc.com

Abstract. All askers who post questions in Community-based Question Answering (CQA) sites such as Yahoo! Answers, Quora or Baidu's Zhidao, expect to receive an answer, and are frustrated when their questions remain unanswered. We propose to provide a type of "heads up" to askers by predicting how many answers, if at all, they will get. Giving a preemptive warning to the asker at posting time should reduce the frustration effect and hopefully allow askers to rephrase their questions if needed. To the best of our knowledge, this is the first attempt to predict the actual number of answers, in addition to predicting whether the question will be answered or not. To this effect, we introduce a new prediction model, specifically tailored to hierarchically structured CQA sites. We conducted extensive experiments on a large corpus comprising 1 year of answering activity on Yahoo! Answers, as opposed to a single day in previous studies. These experiments show that the F_1 we achieved is 24% better than in previous work, mostly due the structure built into the novel model.

1 Introduction

In spite of the huge progress of Web search engines in the last 20 years, many users' needs still remain unanswered. Query assistance tools such as query completion, and related queries, cannot, as of today, deal with complex, heterogeneous needs. In addition, there will always exist subjective and narrow needs for which content has little chance to have been authored prior to the query being issued.

Community-based Question Answering (CQA) sites, such as *Yahoo! Answers*, *Quora*, *Stack Overflow* or *Baidu Zhidao*, have been precisely devised to answer these different needs. These services differ from the extensively investigated Factoid Question Answering that focuses on questions such as "*When was Mozart born?*", for which unambiguous answers typically exist, [1]. Though CQA sites also feature factoid questions, they typically address other needs, such as opinion seeking, recommendations, open-ended questions or very specific needs, *e.g.* "*What type of bird should I get?*" or "*What would you choose as your last meal?*".

Questions not only reflect diverse needs but can be expressed in very different styles, yet, all askers expect to receive answers, and are disappointed otherwise. Unanswered questions are not a rare phenomenon, reaching 13% of the questions in the Yahoo! Answers dataset that we studied, as detailed later, and users whose questions remain

H. Blockeel et al. (Eds.): ECML PKDD 2013, Part III, LNAI 8190, pp. 499–514, 2013.

unanswered are considerably more prone to churning from the CQA service [2] . One way to reduce this frustration is to proactively recommend questions potential answerers, [3,4,5,6]. However, the asker has little or no influence on the answerers' behavior. Indeed, a question by itself may exhibit some characteristics that reduce its potential for answerability. Examples include a poor or ambiguous choice of words, a given type of underlying sentiment, the time of the day when the question was posted, as well as sheer semantic reasons if the question refers to a complex or rare need.

In this work, we focus on the askers, investigate a variety of features they can control, and attempt to predict, based on these features, the expected number of answers a new question might receive, even before it is posted. With such predictions, askers can be warned in advance, and adjust their expectations, if their questions have little chances to be answered. This work represents a first step towards the more ambitious goal of assisting askers in posting answerable questions, by not only indicating the expected number of answers but also suggesting adequate rephrasing. Furthemore, we can imagine additional usages of our prediction mechanism, depending on the site priorities. For instance, a CQA site such as Yahoo! Answers that attempts to satisfy all users might decide to promote questions with few predicted answers in order to achieve a higher answering rate. Alternatively a socially oriented site like Quora, might prefer to promote questions with many predicted answers in order to encourage social interaction between answerers.

We cast the problem of predicting the number of answers as a regression task, while the special case of predicting whether a question will receive any answer at all is viewed as a classification task. We focus on Yahoo! Answers, one of the most visited CQA sites with 30 millions questions and answers a month and 2.4 asked questions per second [7]. For each question in Yahoo! Answers, we generate a set of features that are extracted only from the question attributes and are available before question submission. These features capture asker's attributes, the textual content of the question, the category to which the question is assigned and the time of submission.

In spite of this rich feature set, off-the-shelf regression and classification models do not provide adequate predictions in our tasks. Therefore, we introduce a series of models that better address the unique attributes of our dataset. Our main contributions are threefold:

1. we introduce a novel task of predicting the number of expected answers for a question before it is posted,
2. we devise hierarchical learning models that consider the category-driven structure of Yahoo! Answers and reflect their associated heterogeneous communities, each with its own answering behavior, and finally,
3. we conduct the largest experiment to date on answerability, as we study a year-long question and answer activity on Yahoo! Answers, as opposed to a day-long dataset in previous work.

2 Background

With millions of active users, Yahoo! Answers hosts a very large amount of questions and answers on a wide variety of topics and in many languages. The system is content-centric, as users are socially interacting by engaging in multiple activities around a

specific question. When a user asks a new question, she also assigns it to a specific category, within a predefined hierarchy of categories, which should best match the general topic of the question. For example, the question "*What can I do to fix my bumper?*" was assigned to the category '*Cars & Transportation* > *Maintenance & Repairs*'. Each new question remains "open" for four days (with an option for extension), or less if the asker chose a best answer within this period. Registered users may answer a question as long as it remains "open".

One of the main issues in Yahoo! Answers, and in community-based question answering in general, is the high variance in perceived question and answer quality. This problem drew a lot of research in recent years. Some studies attempted to assess the quality of answers [8,9,10,11], or questions [12,13], and rank them accordingly. Others looked at active users for various tasks such, scoring their "reliability" as a signal for high quality answers or votes [14,15,16], identifying spammers [17], predicting whether the asker of a question will be satisfied with the received answers [18,19] or matching questions to specific users [3,4,5].

Our research belongs to the same general school of work but focuses on estimating the number of answers a question will receive. Prior work that analyzes questions, did it in retrospect, either after the questions had been answered [9], or as a ranking task for a given collection of questions [12,13]. In contrast, we aim at predicting the number of answers for every new question *before* it is submitted.

In a related work, Richardson and White [20] studied whether a question will receive an answer or not. Yet, they conducted their study in a different environment, an IM-based synchronous system, in which potential answerers are known. Given this environment they could leverage features pertaining to the potential answerers, such as reputation. In addition, they considered the specific style of messages sent over IM, including whether a newline was entered and whether some polite words are added. Their experiment was of a small scale on 1,725 questions, for which they showed improvement over the majority baseline. We note that their dataset is less skewed than in Yahoo! Answers. Indeed their dataset counted about 42% of unanswered questions, while Yahoo! Answers datasets typically count about 13% of unanswered questions. We will later discuss the challenges involved in dealing with such a skewed dataset.

A more related prior work that investigated question answerability is Yang et al. [21], who addressed the same task of coarse (yes/no) answerability as above but in the same settings as ours, namely Yahoo! Answers. Yang et al. approached the task as a classification problem with various features ranging from content analysis, such as category matching, polite words and hidden topics, to asker reputation and time of day. They used a one-day dataset of Yahoo! Answers questions and observed the same ratio of unanswered questions as we did in our one-year dataset, namely 13%. Failing to construct a classifier for this heavily skewed dataset, Yang et al. resorted to learning from an artificially balanced training set, which resulted in improvements over the majority baseline. In this paper, we also address this classification task, with the same type of skewed dataset. However, unlike Yang et al., we attempt to improve over the majority baseline without artificially balancing the dataset.

Finally another major differentiator with the above previous work is that we do not stop at simply predicting whether a question will be answered or not, but predict the exact number of answers the question would receive.

3 Predicting Question Answerability

One key requirement of our work, as well as a differentiator with typical prior work on question analysis, is that we want to predict answerability before the question is posted. This imposes constraints on the type of data and signals we can leverage. Namely, we can only use data that is intrinsic to a new question before submission. In the case of Yahoo! Answers, this includes: (a) the title and the body of the question, (b) the category to which the question is assigned, (c) the identity of the user who asked the question and (d) the date and time the question is being posted.

We view the prediction of the expected number of answers as a regression problem, in which a target function (a.k.a the *model*) $\hat{y} = f(x)$ is learned, with x being a vector-space representation of a given question, and $\hat{y} \in \mathbb{R}$ an estimate for y, the number of answers this question will actually receive. All the different models we present in this section are learned from a training set of example questions and their known number of answers, $D = \{(x_i, y_i)\}$. The prediction task of whether a question will receive an answer at all is addressed as a classification task. It is similarly modeled by a target function $\hat{y} = f(x)$ and the same vector space representation of a question, yet, the training target is binary, with answered (unanswered) questions being the positive (negative) examples.

To fully present our models for the two tasks, we next specify how a question representation x is generated, and then introduce for each task novel models (*e.g.* $f(x)$) that address the unique properties of the dataset.

3.1 Question Features

In our approach, each question is represented by a feature vector. For any new question, we extract various attributes that belong to three main types of information: question meta data, question content, and user data. In the rest of this paper we use the term *feature family* to denote a single attribute extracted from the data. Question attributes may be numerical, categorical or set-valued (*e.g.* the set of word tokens in the title). Hence, in order to allow learning by gradient-based methods, we transformed all categorical attributes to binary features, and binned most of the numeric attributes. For example, the category of a question is represented as 1287 binary features and the hour it was posted is represented as 24 binary features. Tables 3.1, 2 and 3 describe the different feature families we extract, grouped according to their information source: the question text, the asker and question meta data.

3.2 Regression Models

Following the description of the features extracted from each question, we now introduce different models (by order of complexity) that use the question feature vector in

Table 1. Features extracted from title and body texts

Feature Family	Description	# Features
Title tokens	The tokens extracted from the title, not including stop words	45,011
Body tokens	The tokens extracted from the body, not including stop words	45,508
Title sentiment	The positive and negative sentiment scores of the title, calculated by the SentiStrength tool [22]	2
Body sentiment	The mean positive and negative sentiment scores of the sentences in the body	2
Supervised LDA	The number of answers estimated by supervised Latent Dirichlet Allocation (SLDA) [23], which was trained over a small subset of the training set	1
Title WH	WH-words (*what*, *when*, *where* ...) extracted from the question's title	11
Body WH	WH-words extracted from the question's body	11
Title length	The title length measured by the number of tokens after stopword removal, binned on a linear scale	10
Body length	The body length, binned on an exponential scale since this length is not constrainted	20
Title URL	The number of URLs that appear within the question title	1
Body URL	The number of URLs that appear within the question body	1

order to predict the number of answers. We remind the reader that our training set consists of pairs $D = \{(x_i, y_i)\}$, where $x_i \in R^F$ is the F dimensional feature vector representation of question q_i, and $y_i \in \{0, 1, 2 \ldots\}$ is the known number of answers for q_i.

Baseline Model. Yang et al. [21] compare the performance of several classifiers, linear and non-linear, on a similar dataset. They report that a linear SVM significantly outperforms all other classifiers. Given these findings, as well as the fact that a linear model is both robust [24] and can be trained very efficiently for large scale problems, we chose a linear model $f(x_i) = w^T x_i + b$ as our baseline model.

Feature Augmentation Model. One of the unique characteristics of the Yahoo! Answers site is that it consists of questions belonging to a variety of categories, each with its community of askers and answerers, temporal activity patterns, jargon etc., and that the categories are organized in a topical taxonomy. This structure, which is inherent to the data, suggests that more complex models might be useful in modeling the data. One effective way of incorporating the category structure of the data in a regression model is to enrich the features with category information. Specifically, we borrowed the idea from [25], which originally utilized such information for domain adaptation.

To formally describe this model, we consider the Yahoo! Answers category taxonomy as a rooted tree T with *"All Categories"* as its root. When referring to the category

Table 2. Features extracted based on the asker

Feature Family	Description	# Features
Asker ID	The identity of the asker, if it asked at least 50 questions in the training set. We ignore askers who asked fewer questions since their ID statistics are unreliable	175,714
Mean # of answers	The past mean number of answers the asker received for her questions, binned on an exponential scale	26
# of questions	The past number of questions asked by the asker, binned on a linear scale and on an exponential scale	26
Log # of questions	The logarithm of the total number of questions posted by the asker in the training set, and the square of the logarithm. For both features we add 1 to the argument of the logarithm to handle test users with no training questions.	2

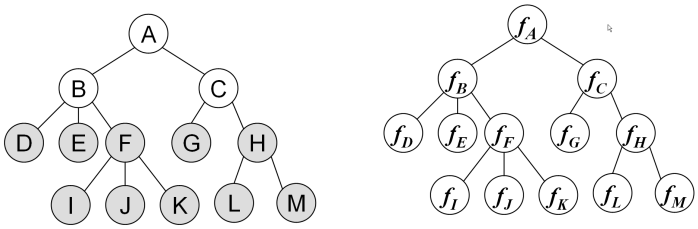
Table 3. Features extracted from the question's meta data

Feature Family	Description	# Features
Category	The ID of the category that the question is assigned to	1,287
Parent Category	The ID of the parent category of the assigned category for the question, based on the category taxonomy	119
Hour	The hour at which the question was posted, capturing daily patterns	24
Day of week	The day-of-week in which the question was posted, capturing weekly patterns	7
Week of year	The week in the year in which the question was posted, capturing yearly patterns	51

tree we will use interchangeably the term *node* and *category*. We denote the category of a question q_i by $C(q_i)$. We further denote by $P(c)$ the set of all nodes on the path from the tree root to node c (including c and the root). For notational purposes, we use a binary representation for $P(c)$: $P(c) \in \{0,1\}^{|T|}$, where $|T|$ is the number of nodes in the category tree.

The feature augmentation model represents each question q_i by $\widehat{x_i} \in R^{F|T|}$ where $\widehat{x_i} = P(C(q_i)) \otimes x_i$ where $\otimes$ represents the Kronecker product. For example, given question q_i that is assigned to category '*Dogs*', the respective node path in T is '*All Questions/Pets/Dogs*'. The feature vector $\widehat{x_i}$ for q_i is all zeros except for three copies of x_i corresponding to each of the nodes '*All Questions*','*Pets*' and '*Dogs*'.

The rationale behind this representation is to allow a separate set of features for each category, thereby learning category specific patterns. These include learning patterns for leaf categories, but also learning lower resolution patterns for intermediate nodes in the tree, which correspond to parent and top categories in Yahoo! Answers. This permits a good tradeoff between high resolution modeling and robustness, obtained by the higher level category components shared by many examples.



(a) Tree of categories (b) Tree of models

Fig. 1. An illustration of the subtree model structure. Shaded nodes in (a) represent categories populated with questions, while unshaded nodes are purely navigational.

Subtree Model. An alternative to the feature augmentation model is to train several linear models, each specializing on a different subtree of T. Let us note a subset of the dataset as $D_c = \{(x_i, y_i) | q_i \in S(c)\}$, where $S(c)$ is the set of categories in the category subtree rooted at node c. We also note a model trained on D_c as f_c. Since there is a one-to-one correspondence between models and nodes in T, the set of models f_c can be organized as a tree isomorphic to T. Models in deeper levels of the tree are specialized on fewer categories than models closer to the root. Figure 1 illustrates a category tree and its corresponding model tree structure. The shaded nodes in 1(a) represent categories to which some training questions are assigned.

One simplistic way of using the model tree structure is to apply the root model (f_A in Figure 1(b)) to all test questions. Note that this is identical to the baseline model. Yet, there are many other ways to model the data using the model tree. Specifically, any set of nodes that also acts as a tree cut defines a regression model, in which the number of answers for a given question q_i is predicted by the first model in the set encountered when traversing from the category c_i, assigned to q_i, to the root of T. In this work, we shall limit ourselves to three such cuts:

TOP: q_i is predicted by model $f_{Top(c_i)}$, where $Top(c)$ is the category in $P(c)$ directly connected to the root.
PARENT: q_i is predicted by model $f_{Parent(c_i)}$
NODE: q_i is predicted by model f_{c_i}

In Figure 1 the TOP model refers to $\{f_B, f_C\}$, the PARENT model refers to $\{f_B, f_C, f_F, f_H\}$ and the NODE model refers to $\{f_D, f_E, \ldots f_M\}$.

Ensemble of Subtree Models. In order to further exploit the structure of the category taxonomy in Yahoo! Answers, the questions in each category c are addressed by all models in the path between this category and the tree root, under the subtree framework described above. Each model in this path introduces a different balance between robustness and specificity. For example, the root model is the most robust, but also the least specific in terms of the idiomatic attributes of the target category c. At the other end of the spectrum, f_c is specifically trained for c, but it is more prone for over fitting the data, especially for categories with few training examples.

Instead of picking just one model on the path from c to the root, the ensemble model for c learns to combine all subtree models by training a meta linear model:

$$f(x_i) = \sum_{c' \in P(c)} \alpha_{cc'} f_{c'}(x_i) + b_c \quad (1)$$

where $f_{c'}$ are the subtree models described previously and the weights $\alpha_{cc'}$ and b_c are optimized over a validation set. For example, the ensemble model for questions assigned to category E in Figure 1(a) are modeled by a linear combination of models f_A, f_B and f_E, which are trained on training sets D, D_B and D_E respectively.

3.3 Classification Models

The task of predicting whether a question will be answered or not is an important special case of the regression task. In this classification task, we treat questions that were not answered as negative examples and questions that were answered as the positive examples. We emphasize that our dataset is skewed, with the negative class constituting only 12.68% of the dataset. Furthermore, as already noted in [26], the distribution of the number of answers per question is very skewed, with a long tail of questions having high number of answers.

As described in the background section, this task was studied by Yang et al. [21], who failed to provide a solution for the unbalanced dataset. Instead, they artificially balanced the classes in their training set by sampling, which may reduce the performance of the classifier on the still skewed test set. Unlike Yang et al., who used off-the-shelf classifiers for the task, we devised classifiers that specifically address the class imbalance attribute of the data. We noticed that a question that received one or two answers could have easily gone unanswered, while this is unlikely for questions with dozen answers or more. When projecting the number of answers y_i into two values, this difference between positive examples is lost and may produce inferior models. The following models attempt to deal with this issue.

Baseline Model. Yang et al. [21] found that linear SVM provides superior performance on this task. Accordingly, we choose as baseline a linear model, $f(x_i) = w^T x_i + b$ trained with hinge loss. We train the model on the binarized dataset $D_0 = \{(x_i, \text{sign}(y_i - 1/2))\}$ (see our experiment for more details).

Feature Augmentation Model. In this model, we train the same baseline classifier presented above. Yet the feature vector fed into the model is the augmented feature representation introduced for the regression models.

Ensemble Model. In order to capture the intuition that not "all positive examples are equal", we use an idea closely related to works based on Error Correcting Output Coding for multi-class classification [27]. Specifically, we construct a series of binary classification datasets $D_t = \{(x_i, z_i^t)\}$ where $z_i^t = \text{sign}(y_i - 1/2 - t)$ and $t = 0, 1, 2, \ldots$. In this series, D_0 is a dataset where questions with one or more answers are considered

positive, while in D_{10} only examples with more than 10 answers are considered positive. We note that these datasets have varying degrees of imbalance between the positive and the negative classes.

Denoting by f_t the classifier trained on D_t, we construct the final ensemble classifier by using a logistic regression

$$f(x) = \sigma(\sum_t \alpha_t f_t(x) + b) \tag{2}$$

where $\sigma(u) = (1 + e^{-u})^{-1}$ and the coefficients α_t and b are learned by minimizing the log-likelihood loss on the validation set.

Ensemble of Feature Augmentation Models. In this model, we train the same ensemble classifier presented above. Yet the feature vector fed into the model is the augmented feature representation introduced for the regression models.

Classification Ensemble of Subtree Models. As our last classification model, we directly utilize regression predictions to differentiate between positive examples. We use the same model tree structure used in the regression by ensemble of subtree models. All models are linear regression models trained exactly as in the regression problem, in order to predict the number of answers for each question. The final ensemble model is a logistic regression function of the outputs of the individual regression models:

$$f(x_i) = \sigma(\sum_{c' \in P(c)} \alpha_{cc'} f_{c'}(x_i) + b_c) \tag{3}$$

where $f_{c'}$ are the subtree regression models and the weights $\alpha_{cc'}$ and b_c are trained using the validation set.

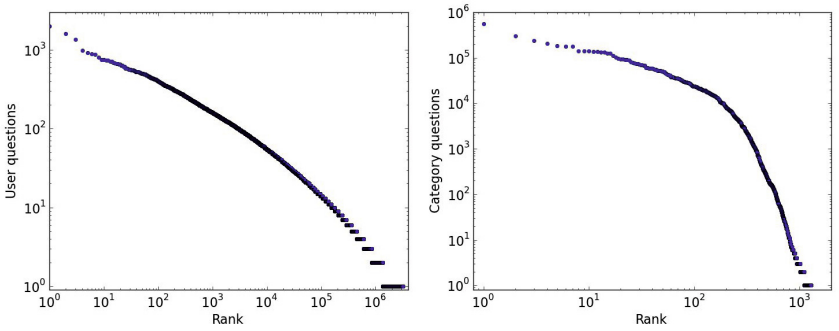
4 Experiments

We describe here the experiments we conducted to test our regression and classification models, starting with our experimental setup, then presenting our results and analyses.

4.1 Experimental Setup

Our dataset consists of a uniform sample of 10 million questions out of all non-spam English questions submitted to Yahoo! Answers in 2009. The questions in this dataset were asked by more than 3 million different users and were assigned to $1,287$ categories out of the $1,569$ categories. A significant fraction of the sampled questions (12.67%) remained unanswered. The average number of answers per question is 4.56 ($\sigma = 6.11$). The distribution of the number of answers follows approximately a geometric distribution.

The distributions of questions among users and among categories are extremely skewed, with a long tail of users who posted one or two questions and sparsely populated categories. These distributions are depicted in Figure 2, showing a power law



(a) Questions per user (b) Questions per category

Fig. 2. Distribution of number of questions depicted as a function of ranks

behavior for the questions per asker distribution. A large fraction of the categories have quite a few questions, for example, about half of all categories in our dataset count less than 50 examples.

We randomly divided our dataset into three sets: 80% training, 15% test and 5% validation (for hyper-parameter tuning). Very few questions in the dataset attracted hundreds of answers. To eliminate the ill effect of these questions on model training, we modified the maximum number of answers per question to 64. This resulted in changing the target of about 0.03% of the questions.

Due to its speed, robustness and scalability, we used the Vowpal Wabbit tool[1] whenever possible. All regression models were trained with squared loss, except for ensemble of subtree models, Eq. 1, whose coefficients were learned by a least squares fit. All classification models were trained using Vowpal Wabbit with hinge loss, except for the ensemble models, Eq. 2 and 3, whose coefficients were learned by maximizing the log-likelihood of the validation set using Stochastic Gradient Descent. We note that for a node c, where ensemble models should have been trained based on less than 50 validation examples, we refrained from training the ensemble model and used the NODE subtree model of c as a single component of the ensemble model.

Table 4 compares between the various trained models with respect to the number of basic linear models used in composite models and the average number of features observed per linear model. The meta-parameters of the ensemble models (Eq. 1 and 3) were not included in the counting.

4.2 Results

The performance of the different regression models on our dataset was measured by Root Mean Square Error (RMSE) [28] and by the Pearson correlation between the predictions and the target. Table 5 presents these results. As can be seen, all our models outperform the baseline off-the-shelf linear regression model, with the best performing

[1] `http://hunch.net/~vw/`

Table 4. Details of the regression and classification models, including the number of linear models in each composite model, and the average number of features used by each linear model

Regression		
Model	# linear models	features per model
Baseline	1	267,781
Feature augmentation	1	12,731,748
Subtree - TOP	26	88,358
Subtree - PARENT	119	26,986
Subtree - NODE	924	9,221
Ens. of subtree models	955	13,360

Classification		
Model	# linear models	features per model
Baseline	1	267,781
Feature augmentation	1	12,731,748
Ensemble	7	267,781
Feature augmentation Ens.	7	12,731,748
Ens. of subtree models	955	13,360

Table 5. Test performance for the regression models

Model	RMSE	Pearson Correlation
Baseline	5.076	0.503
Feature augmentation	4.946	0.539
Subtree - TOP	4.905	0.550
Subtree - PARENT	4.894	0.552
Subtree - NODE	4.845	0.564
Ens. of subtree models	4.606	0.620

Table 6. Test performance for the classification models

Model	AUC
Baseline	0.619
Feature augmentation	0.646
Ensemble	0.725
Feature augmentation ensemble	0.739
Ensemble of subtree models	0.781

model achieving about 10% relative improvement. These results indicate the importance of explicitly modeling the different answering patterns within the heterogeneous communities in Yahoo! Answers, as captured by categories. Interestingly, the feature-augmentation model, which attempts to combine between categories and their ancestors, performs worse than any specific subtree model. One of the reasons for this is the huge number of parameters this model had to train (see Table 4), compared to the ensemble of separately trained subtree models, each requiring considerably fewer parameters to tune. A t-test based on the Pearson correlations shows that each model in Table 5 is significantly better than the preceding one, with P-values close to zero.

The performance of models for the classification task was measured by the area under the ROC Curve (AUC) [29]. AUC is a preferred performance measure when class distributions are skewed, since it measures the probability that a positive example is scored higher than a negative example. Specifically, the AUC of a majority model is always 0.5, independently of the distribution of the targets.

Inspecting the classification results in Table 6, we can see that all the novel models improve over the baseline classifier, with the best performing ensemble of subtrees classifier achieving an AUC of 0.781, a substantial relative improvement of 26% over the baseline's result of 0.619. A t-test based on the estimated variance of AUC [30] shows that each model in Table 6 is statistically significantly superior to its predecessor with P-values practically zero.

We next examine in more depth the performance of the ensemble of classifiers and the ensemble of subtree regressors (the third and fifth entries in Table 6 respectively). We see that the ensemble of classifiers explicitly models the differences between questions with many and few answers, significantly improving over the baseline. Yet, the ensemble of subtree regressors not only models this property of the data but also the differences in answering patterns within different categories. Its higher performance indicates that both attributes are key factors in prediction. Thus the task of predicting the actual number of answers has additional benefits, it allows for a better understanding of the structure of the dataset, which also helps for the classification task.

Finally, we compared our results to those of Yang et al. [21]. They measured the F_1 value on the predictions of the minority class of unanswered questions, for which their best classifier achieved an F_1 of 0.325. Our best model for this measure was again the ensemble of subtree models classifier, which achieved an F_1 of 0.403. This is a substantial increase of 24% over Yang et al.'s best result, showing again the benefits of a structured classifier.

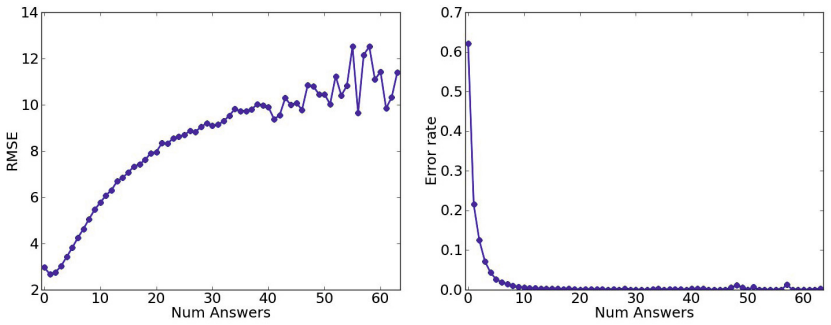
4.3 Error Analysis

We investigated where our models err by measuring the average performance of our best performing models as a function of the number of answers per test question, see Figure 3. We split the test examples into disjoint sets characterized by a fixed number of answers per question and averaged the RMSE of our best regressor on each set (Figure 3(a)). Since our classifier is not optimized for the Accuracy measure, we set a specific threshold on the classifier output, choosing the 12.67[2] percentile of test examples with lowest scores as negatives. Figure 3(b) shows the error rate for this threshold. We note that the error rate for zero answers refers to false positives rate and for all other cases it refers to the false negatives rate.

Figure 3(a) exhibits a clear minimum in the region most populated with questions, which shows that the regressor is optimized for predicting values near 0. Although the RMSE increases substantially with the number of answers, it is still moderate. In general, the RMSE we obtained is approximately linear to the square root of the number of answers. Specifically, for questions with large number of answers, the RMSE is much smaller than the true number of answers. For example, for questions with more than 10 answers, which constitute about 13% of the dataset, the actual number of answers is approximately twice the RMSE on average. This shows the benefit of using the regression models as input to a answered/unanswered classifier, as we did in our best performing classifier. This is reflected, for example, in the very low error rates (0.0064 or less) for questions with more than 10 answers in Figure 3(b).

While the regression output effectively directs the classifier to the correct decision for questions with around 5 or more answers, Figure 3(b) still exhibits substantial error rates for questions with very few or no answers. This is due to the inherent randomness in the answering process, in which questions that received very few answers could have easily gone unanswered and vice versa and are thus difficult to predict accurately.

[2] This is the fraction of negative examples in our training set.



(a) Regression task (b) Classification task

Fig. 3. Performance of the Subtree Ensemble models as a function of the number of answers

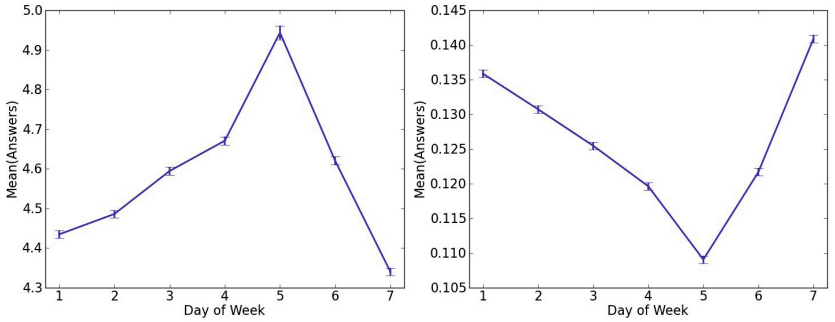
In future work, we want to improve these results by employing boosting approaches and constructing specialized classifiers for questions with very few answers.

4.4 Temporal Analysis

Intuitively, the time at which a question is posted should play a role in a social-media site, therefore, like Yang et al. [21], we use temporal features. Our dataset, which spans over one year, confirms their reported patterns of hourly answering: questions posted at night are most likely to be answered, while "afternoon questions" are about 40% more likely to remain unanswered.

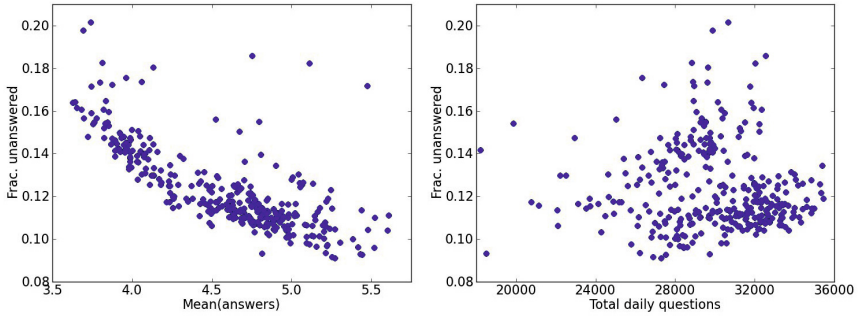
To extend this analysis to longer time periods, we analyzed weekly and yearly patterns. We first calculated the mean number of answers per question and the fraction of unanswered questions as a function of the day of week, as shown in Figure 4. A clear pattern can be observed: questions are more often answered towards the end of the week, with a sharp peak on Fridays and a steep decline over the weekend. The differences between the days are highly statistically significant (t-test, two sided tests). The two graphs in Figure 4 exhibit extremely similar characteristics, indicating that the fraction of unanswered questions is negatively correlated with the average number of answers per question. This suggests that both phenomena are controlled by a supply and demand equilibrium. This can be explained by two hypotheses: (a) both phenomena are driven by an increase in questions (Yang et al.'s hypothesis) or (b) both phenomena are driven by a decrease in the number of answers.

To test the above two hypotheses, we extracted the number of questions, number of answers and fraction of unanswered questions on a daily basis. Each day is represented in Figure 5 as a single point, as we plot the daily fraction of unanswered questions as a function of the daily average number of answers per question (Figure 5(a)) and as a function of the total number of daily questions (Figure 5(b)). We note that while some answers are provided on a window of time longer than a day, this is a rare phenomenon. The vast majority of answers are obtained within about twenty minutes from the question posting time [5], hence our daily analysis.



(a) Mean answers (b) Frac. unanswered

Fig. 4. Mean number of answers and fraction of number of answers as a function of the day of the week, where '1' corresponds to Monday and '7' to Sunday



(a) (b)

Fig. 5. The daily fraction of unanswered questions as a function of the daily mean number of answers and as a function of the total number of questions

Figure 5(a) exhibits a strong negative correlation (Pearson correlation $r = -0.631$), while almost no effect is observed in Figure 5(b) ($r = 0.010$). We further tested the correlation between the daily total number of answers and the fraction of fraction of unanswered questions, and here as well a significant negative correlation was observed ($r = -0.386$). These findings support the hypothesis that deficiency in answerers is the key factor affecting the fraction of unanswered questions, and not the overall number of questions, which was Yang et al's hypothesis. This result is important, because it implies that more questions in a community-based question answering site will not reduce the performance of the site, as long as an active community of answerers strives at its core.

5 Conclusions

In this paper, we investigated the answerability of questions in community-based question answering sites. We went beyond previous work that returned a binary result of

whether or not the question will be answered. We focused on the novel task of predicting the actual number of expected answers for new questions in community-based question answering sites, so as to return feedback to askers before they post their questions. We introduced a series of novel regression and classification models explicitly designed for leveraging the unique attributes of category-organized community-based question answering sites. We observed that these categories host diverse communities with different answering patterns.

Our models were tested over a large set of questions from Yahoo! Answers, showing significant improvement over previous work and baseline models. Our results confirmed our intuition that predicting answerability at a finer grained level is beneficial. They also showed the strong effect of the different communities interacting with questions on the number of answers a question will receive. Finally, we discovered an important and somehow counter-intuitive fact, namely that an increased number of questions will not negatively impact answerability, as long as the community of answerers is maintained.

We constructed models that are performant at scale: even the ensemble models are extremely fast at inference time. In future work, we intend to increase response time even further and consider incremental aspects in order to return predictions as the asker types, thus providing, in real-time, dynamic feedback and a more engaging experience. To complement this scenario, we are also interested in providing question rephrasing suggestions for a full assistance solution.

References

1. Voorhees, E.M., Tice, D.M.: Building a question answering test collection. In: SIGIR (2000)
2. Dror, G., Pelleg, D., Rokhlenko, O., Szpektor, I.: Churn prediction in new users of yahoo! answers. In: WWW (Companion Volume), pp. 829–834 (2012)
3. Li, B., King, I.: Routing questions to appropriate answerers in community question answering services. In: CIKM, pp. 1585–1588 (2010)
4. Horowitz, D., Kamvar, S.D.: The anatomy of a large-scale social search engine. In: Proceedings of the 19th International Conference on World Wide Web, WWW 2010, pp. 431–440. ACM, New York (2010)
5. Dror, G., Koren, Y., Maarek, Y., Szpektor, I.: I want to answer; who has a question?: Yahoo! answers recommender system. In: KDD, pp. 1109–1117 (2011)
6. Szpektor, I., Maarek, Y., Pelleg, D.: When relevance is not enough: Promoting diversity and freshness in personalized question recommendation. In: WWW (2013)
7. Rao, L.: Yahoo mail and im users update their status 800 million times a month. TechCrunch (October 28, 2009)
8. Jeon, J., Croft, W.B., Lee, J.H., Park, S.: A framework to predict the quality of answers with non-textual features. In: Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2006, pp. 228–235. ACM, New York (2006)
9. Agichtein, E., Castillo, C., Donato, D., Gionis, A., Mishne, G.: Finding high quality content in social media, with an application to community-based question answering. In: Proceedings of ACM WSDM, WSDM 2008. ACM Press, Stanford (2008)
10. Shah, C., Pomerantz, J.: Evaluating and predicting answer quality in community qa. In: Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2010, pp. 411–418. ACM, New York (2010)

11. Surdeanu, M., Ciaramita, M., Zaragoza, H.: Learning to rank answers on large online qa collections. In: ACL, pp. 719–727 (2008)
12. Song, Y.I., Lin, C.Y., Cao, Y., Rim, H.C.: Question utility: A novel static ranking of question search. In: AAAI, pp. 1231–1236 (2008)
13. Sun, K., Cao, Y., Song, X., Song, Y.I., Wang, X., Lin, C.Y.: Learning to recommend questions based on user ratings. In: CIKM, pp. 751–758 (2009)
14. Jurczyk, P., Agichtein, E.: Discovering authorities in question answer communities by using link analysis. In: Proceedings of the Sixteenth ACM Conference on Information and Knowledge Management, CIKM 2007, pp. 919–922. ACM, New York (2007)
15. Bian, J., Liu, Y., Zhou, D., Agichtein, E., Zha, H.: Learning to recognize reliable users and content in social media with coupled mutual reinforcement. In: WWW, pp. 51–60 (2009)
16. Lee, C.T., Rodrigues, E.M., Kazai, G., Milic-Frayling, N., Ignjatovic, A.: Model for voter scoring and best answer selection in community q&a services. In: Web Intelligence, pp. 116–123 (2009)
17. Lee, K., Caverlee, J., Webb, S.: Uncovering social spammers: social honeypots + machine learning. In: SIGIR, pp. 435–442 (2010)
18. Liu, Y., Agichtein, E.: You've got answers: Towards personalized models for predicting success in community question answering. In: ACL (Short Papers), pp. 97–100 (2008)
19. Agichtein, E., Liu, Y., Bian, J.: Modeling information-seeker satisfaction in community question answering. ACM Transactions on Knowledge Discovery from Data 3(2), 10:1–10:27 (2009)
20. Richardson, M., White, R.W.: Supporting synchronous social q&a throughout the question lifecycle. In: WWW, pp. 755–764 (2011)
21. Yang, L., Bao, S., Lin, Q., Wu, X., Han, D., Su, Z., Yu, Y.: Analyzing and predicting not-answered questions in community-based question answering services. In: AAAI (2011)
22. Thelwall, M., Buckley, K., Paltoglou, G., Cai, D., Kappas, A.: Sentiment in short strength detection informal text. J. Am. Soc. Inf. Sci. Technol. 61(12), 2544–2558 (2010)
23. Blei, D., McAuliffe, J.: Supervised topic models. In: Platt, J., Koller, D., Singer, Y., Roweis, S. (eds.) Advances in Neural Information Processing Systems 20. MIT Press, Cambridge (2008)
24. Draper, N.R., Smith, H.: Applied Regression Analysis, 3rd edn. Wiley Series in Probability and Statistics. Wiley-Interscience (April 1998)
25. Daume III, H.: Frustratingly easy domain adaptation. In: Proceedings of the 45th Annual Meeting of the Association of Computational Linguistics, Prague, Czech Republic, pp. 256–263. Association for Computational Linguistics (June 2007)
26. Adamic, L.A., Zhang, J., Bakshy, E., Ackerman, M.S.: Knowledge sharing and yahoo answers: everyone knows something. In: Proceedings of the 17th International Conference on World Wide Web, WWW 2008, pp. 665–674. ACM, New York (2008)
27. Dietterich, T.G., Bakiri, G.: Solving multiclass learning problems via error-correcting output codes. Journal of Artificial Intelligence Research 2 (1995)
28. Bibby, J., Toutenburg, H.: Prediction and Improved Estimation in Linear Models. John Wiley & Sons, Inc., New York (1978)
29. Provost, F.J., Fawcett, T.: Analysis and visualization of classifier performance: Comparison under imprecise class and cost distributions. In: KDD, pp. 43–48 (1997)
30. Cortes, C., Mohri, M.: Confidence intervals for the area under the roc curve. In: NIPS (2004)