Predicting Answering Behaviour in Online Question Answering Communities

Predicting Answering Behaviour in Online Question Answering Communities

Grégoire Burel
Knowledge Media institute
Open University, UK
gregoire.burel@open.ac.uk

Paul Mulholland
Knowledge Media institute
Open University, UK
paul.mulholland@open.ac.uk

Yulan He
School of Engineering & Applied Science
Aston University, UK
y.he@cantab.net

Harith Alani
Knowledge Media institute
Open University, UK
h.alani@open.ac.uk

ABSTRACT

The value of Question Answering (Q&A) communities is dependent on members of the community finding the questions they are most willing and able to answer. This can be difficult in communities with a high volume of questions. Much previous has work attempted to address this problem by recommending questions similar to those already answered. However, this approach disregards the question selection behaviour of the answers and how it is affected by factors such as question recency and reputation. In this paper, we identify the parameters that correlate with such a behaviour by analysing the users' answering patterns in a Q&A community. We then generate a model to predict which question a user is most likely to answer next. We train Learning to Rank (LTR) models to predict question selections using various *user*, *question* and *thread* feature sets. We show that answering behaviour can be predicted with a high level of success, and highlight the particular features that influence users' question selections.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

Keywords

social Q&A platforms; online communities; user behaviour; social media

1. INTRODUCTION

Online Question Answering (Q&A) communities such as Stack Exchange,[1] Yahoo! Answers,[2] and Quora,[3] have witnessed a rapid increase in popularity in recent years as more and more people are going online to seek answers to their questions.

In Q&A communities, such as Stack Exchange, more than 3,700 new questions are posted every day.[4] As a result, users need effective methods to help them find the questions they are interested in, or capable of, answering. In this context, much research has focused on question routing [23, 10, 11, 9, 5]; the automatic identification of relevant potential answerers for a given question. Another approach is user-centric question recommendation [12], where the focus is on identifying the most suitable question to a given potential answerer.

All such techniques aim at more efficiently connecting questions and potential answerers to boost community reactivity. Such techniques are largely based on matching the user's topic affinity and expertise with available questions, often overlooking the possible influence of users' behavioural patterns and preferences on their question selections. In this paper, we extract such behavioural patterns and use them to predict the questions that each answerer will select in the Stack Exchange *Cooking* community (CO);[5] which is a food oriented Q&A website. The main contributions of this paper are:

1. Demonstrate a method for extracting patterns of question-selection behaviour of individual users in a Q&A community.
2. Study the influence of 62 *user*, *question*, and *thread* features on answering behaviour and show how combining these features increases the quality of behaviour predictions.
3. Investigate the use of Learning to Rank models (LTR) for identifying the most relevant question for a user at any given time.

[1] `http://stackexchange.com/`.
[2] `http://answers.yahoo.com/`.
[3] l`https://www.quora.com/`.
[4] l`https://api.stackexchange.com/docs/info#filter=default&site=stackoverflow&run=true`
[5] `http://cooking.stackexchange.com`.

HT '15, September 1–4, 2015, Guzelyurt, Northern Cyprus.

DOI: http://dx.doi.org/10.1145/2700171.2791041

4. Construct multiple models to predict question-selections, and compare against multiple baselines (question recency, topic affinity, and random), achieving high precision gains (+93%).

In the following section we discuss existing works in question recommendation, question routing and answering behaviour. In Section 3 we introduce our approach for extracting and representing answering behaviour and discuss the set of features used for building our question-selection prediction model. In Section 4, we present our experiment and results. Discussions and conclusions are provided in Sections 5 and 6 respectively.

2. RELATED WORK

Question routing and recommendation can be seen as a sub-branch of content recommendation, the task of recommending items based on item description and user interests [14]. Since the goal of recommendation systems is to identify what content is likely to interest users, it shares many similarities with behaviour modelling. In the context of Q&A, question routing is generally contrasted from question recommendation as the task of identifying potential answerers for a given question whereas question recommendation is typically restricted to the recommendation of questions given a query or question (e.g. retrieval of similar questions). A slightly different task is answerer-centric question recommendation, the task of recommending questions to potential answerers.

Although question routing [22, 15, 23, 10, 12, 11, 6, 17, 9, 5, 20] and question recommendation [3, 7] research aim at different problems, both approaches have similar methodologies. Question recommendation approaches can be easily used for question routing and vice versa.

Most research has focused on the computation of language or topic similarity between questions and users in order to decide how relevant is a user given a question. Many works rely on topic models or similar approaches [22, 23, 15, 10, 17, 5, 20, 21] as part of the features used for classifying user as relevant for a question.

In general, the majority of works have relied on supervised binary classifiers trained on different features that can classify questions as relevant and irrelevant. Although providing good recommendation results, such methods do not necessary take into account relations between recommended items due to the type of algorithms used. Moreover, previous techniques are normally insensitive to the constant change in the community, available questions, and the status of those questions. For example, a question may receive a new answer from the community and render it less attractive to other potential answerers, or a new relevant question posed by a reputable user could divert the attention of potential answerers. Capturing such dynamic behaviour patterns impose the need for more complex approaches. In order to account for the relations between the content to be recommended and provide better recommendations, [12] and [19] used LTR models as part of their classifier. Although better than binary models, such approaches still considered non-dynamic features and do not consider the evolution of communities.

Besides routing the questions that fit user needs or topics and other standard text features, works like [10] and [5] integrated some behaviour factors in the recommendation process, such as user performance and availability in order to maximise the chances of getting good answers. To some extent, these works can be related with best answer identification within a thread such as [2] and [21]. Nevertheless, these methods also mostly focused on non dynamic features or where not trying to directly model answerer behaviour.

Few works in this domain have also looked at graph based techniques. However, rather than focusing on user decisions, such models focused on user reputation graphs [9] or topic hierarchies [3].

Our approach differs from all the above for three main reasons:

1. We use a mixture of dynamically-calculated *question*, *thread* and *user* (potential answerer) features.

2. We consider *all available questions* at each contribution time rather than only recently posed questions.

3. We identify which features correlate the most with user behaviour.

3. MODELLING QUESTION SELECTION BEHAVIOUR IN Q&A COMMUNITIES

Identifying the questions that a user will answer requires the definition of the choices and activities that the user is presented with at any given point in time. Once these choices and user behaviours are modelled, we can then construct a method to automatically predict question-selections based on available choices (i.e. predicting the correct user action given a set of possible outcomes).

3.1 Answerer Behaviour in Q&A Websites

In this section we describe some typical answering dynamics in Q&A websites, which naturally influence and guide answerers' behaviour. Q&A websites tend to have similar designs and contribution mechanisms. Such systems apply various ranking algorithms to present users with a list of questions that need answering. For example, Stack Exchange uses the number of community votes by default to rank questions, whereas Yahoo! Answers tends to rank questions based on their recency. Most Q&A websites enable users to flag best answers, and use this information to halt promoting those questions further. Users are usually encouraged to update their answers rather than posting new ones to the same questions. In the community we analyse in this paper, we found no cases where a user replied more than once to the same question.

On a typical Q&A website, a user is presented with a list of available and open questions to browse and select one question at a time to answer. Once an answer is provided, the question is dropped from the list of available questions, which will be given back to that user to select further questions to answer. In summary, the answering behaviour of users can be broadly divided into a two-step iterative process:

1. Obtain the list of available questions.

2. Select a question and answer it.

Next we try to model these steps, along with the features that could influence them.

3.2 Representing Answerer Behaviour

In order to introduce the method used for learning the behaviour of answerers, we need to formally represent the

different actions and choices a user can make at interaction time, as well as the different factors that a user may take into account when deciding which question to select.

A question $q \in Q$ is characterised by two statuses that change over time, from opened or available ($\mathcal{O}$) to closed or solved ($\mathcal{C}$). We denote the status of a question q at time $t \in T$ as $S : T \times Q \rightarrow \{\mathcal{O}, \mathcal{C}\}$ $(S(q,t))$. We also define the status of question in relation to a particular user $u \in U$ at time t as $S_U : T \times Q \times U \rightarrow \{\mathcal{O}, \mathcal{C}\}$ $(S_U(q,t,u))$ in order to account for the potential previous answer of a user to a question. When a user answers a question, its status will be closed even if the question is not yet marked as solved by the community.

Using the previous notation, the set of available questions $Q_{t_u} = \{q_1, q_2, \ldots, q_n\}$ at t for user u can be defined as function $A : T \times U \rightarrow Q'$ where Q' $(A(t,u))$ is a set of non overlapping elements of Q. For a set of available questions Q_{t_u} at time t for user u, a user selects one question to answer within Q_{t_u}. Such a selection can be defined as $C : Q' \times U \rightarrow q \in Q'$ $(C(Q',u))$. The decision to pick a particular question over a set of available questions can be represented as a vector of decision labels over questions. By combining all such vectors we obtain a time indexed matrix for each user containing all the questions $q \in Q$ and the selection and status of each question. At a particular activity time $t \in T$ for a particular user $u \in U$, an answering choice can be represented as a Boolean where only one question is labelled as selected.

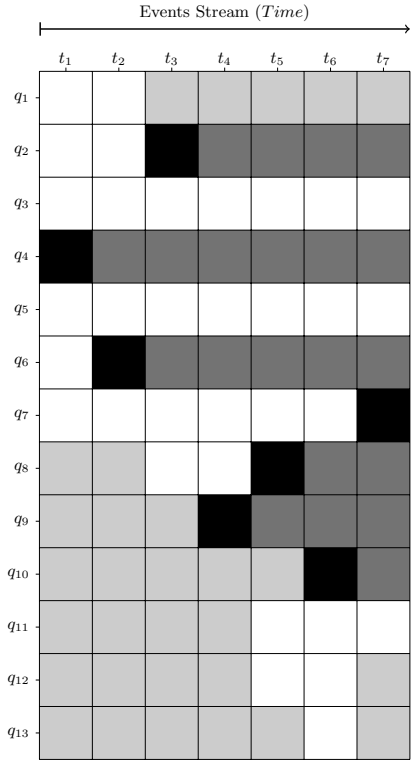
By using such a matrix, a model of a particular user behaviour can be constructed (Figure 1a). For each matrix column (i.e. activity time), a decision graph representing the decision relations between the selected question and the unselected available ones can be constructed (Figure 1b). By using such a representation, we obtain a partially ordered set. Such a graph can then be used for training learning algorithms to automatically determine the answering behaviour of Q&A community members (Section 3.4).

3.3 Features for Predicting Question Selections

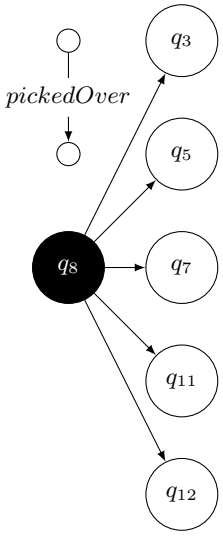
Many different types of features can influence users' selection of questions to answer. We divide such features into question (F_Q), thread (F_T) and user (F_O) categories. Question features are intrinsic to the content of the question itself, whereas Thread features represent all the answers given to a particular question. User features capture the status of the answerer at answering time, and measure affinity between a user and a considered question. It is important to note that all these features evolve during the lifetime of a question as new answers are posted from different users.

For a given user $u \in U$, time $t \in T$ and question $q \in Q$, the features that represent a question are defined by $F : Q \times T \times U \rightarrow F_Q \times F_T \times F_O$ $(F(q,t,u))$. Using the notation used in the previous section, a decision function $D : Q' \times F \rightarrow Q'$ $(D(Q',F))$ can be used for deciding which question to select for every answering activity. As a consequence, understanding how users select questions to answer could be achieved by modelling $D(Q',F)$ and learning the parameters that identify selected questions.

Below we describe 62 features used in this paper, which are strictly generated from the information available at the time when a user selected a question to answer (i.e. future information are not taken into account when calculating those features).



(a) Matrix-like representation of question statuses and answering behaviour for a user.



(b) Decision graph for a user at answering time t_5.

Figure 1: (a) is matrix-like representation of the behaviour of a user $u \in U$ with 7 answering activities out of a total of up to 13 questions $Q = \{q_1, q_2, q_3, \cdots, q_{13}\}$. (b) is a decision graph representing which question is favoured at time t_5 by u. Grey areas are unavailable questions (i.e. not opened yet, already selected, or solved). Black areas represent users selections. White areas are open questions.

Although many features can be potentially selected for predicting users' selection of questions to answer, we decide to use features similar to the ones we have explored previously while investigating the issue of best answer identification [2] and the measurement question complexity [1]

3.3.1 User Features

Features of users who are considering answering a question.

- *Number of Answers*: The number of answers posted by the user so far.
- *Reputation*: Aggregation of user's contribution ratings.
- *Answering Success*: Number of user's previous answers that were flagged as best answers.
- *Number of Posts*: Number of questions and answers previously posted by the user.
- *Number of Questions*: Total number of questions posted by the user until now.
- *Question Reputation*: Represents how liked are the user's questions based on community ratings.
- *Answer Reputation*: Represents how liked are the user's answers based on community ratings.
- *Asking Success*: Number of previous user questions that were identified as solved.
- *Topic Reputation*: Measures user's reputation for a given question. It is derived from the topics or tags[6] T_q associated with a question q. Given a user u, and a question q with a set of topics T_q, the reputation associated with a given question q for user u is given by summing up the score values of user u's previous answer A_u about a particular topic $t \in T_q$:

$$E_P(q,u) = \sum_{t \in T_q} \sum_{a \in A_u} S(a^t) \tag{1}$$

- *Topic Affinity*: Represents the likelihood of a user to answer a question with similar topics. It is calculated from the intersection $t \in T_u \cap T_q$ between the topics or tags T_u of a user u and the topics or tags T_q associated with a potential question q and the probability of u to contribute to each of those topics $P(t|u)$. Given a user u, the user topic affinity associated with a question q is given by:

$$A_u(T_q) = \prod_{t \in T_u \cap T_q} P(t|u) \tag{2}$$

We also include the following features which we derive from the list above: *average answer reputation*, *average question reputation*, *ratio of successfully answered questions*, *ratio of successfully solved questions*, *average observer reputation*, *ratio of reputation for a potential question*, and *average topic reputation*. Total number of User Features is therefore 17.

[6]For the dataset used in this paper (CO), we use the community assigned tags attached to each questions.

3.3.2 Question Features

- *Question Age*: No. of days since question creation.
- *Number of Words*: Number of words in the question.
- *Referral Count*: Number of hyperlinks in the question.
- *Readability with Gunning Fog Index*: Measures question readability using the Gunning index of a question q which is calculated using the average sentence length asl_q and the percentage of complex words pcw_q:

$$G_q(asl_q, pcw_q) = 0.4\ (asl_q + pcw_q) \tag{3}$$

- *Readability with LIX*: Measures readability using the LIX metric of a question q which is calculated using the number of words contained in the question w_q, the number of periods, colon or capital first letters c_q and the number of words with more than six letters $w_q^{>6}$:

$$LIX_q(w_q, c_q, w_q^{>6}) = \frac{w_q}{c_q} \times \frac{w_q^{>6}}{w_q} \times 100 \tag{4}$$

- *Cumulative Term Entropy (Complexity)*: Represents the distribution of words in a question using cumulative entropy. Given question q, its vocabulary V_q, and the frequency of each word $f_w, w \in \{1, 2, ..., |V_q|\}$, cumulative term entropy $C_d(q)$ can be defined as:

$$C_d(q) = \sum_{w=1}^{|V_q|} \frac{f_w \times (\log |V_q| - \log f_w)}{|V_q|} \tag{5}$$

- *Question Polarity*: Measures the average polarity of a question using SentiWordNet[7]. It is calculated from each unique word w contained in a given question q, $w \in \{1, 2, ..., |V_q|\}$, and the positive $pos(w)$ and negative $neg(w)$ weights of word w:

$$QP_q = \frac{1}{|V_q|} \times \sum_{w=1}^{|V_q|} pos(w) - neg(w) \tag{6}$$

Additionally, we include **asker features**, which are the user's features but applied to the asker of the each question. The aim is to take into account the possible influence of the user who posed a question, on the probability of that question being selected by others for answering. Therefore each question has also all the 17 user features, calculated for each question asker, thus totalling 23 Question Features.

3.3.3 Thread Features

This feature set aggregates the features of all the answers already posted to a question by the time a user is selecting a question to answer.

Each thread feature (F_T) is calculated using the same question features above, apart from *question age*, and normalised (i.e. averaged) across all the questions posted at any given time. Total number of Thread Features is 22.

[7]SentiWordNet, `http://sentiwordnet.isti.cnr.it/`.

3.4 Learning Behaviour using LTR Models

The question-selection function can be seen as a Learning to Rank problem where a learning algorithm tries to generate a list of ranked items based on derived relevance labels. In our case, the goal is to find the question that is most likely to be answered by a particular user. In other words, for each selection time, we try to label one question from the list as relevant, and label all the others as irrelevant. Such a decision representation can be built from the decision graphs discussed in Section 3.2.

One simple approach is to use a *pointwise* LTR method [13], which applies standard classifiers to rank questions directly by only considering them individually and then ordering them. For a binary relevance label, one approach is to use regression models and then rank the results by the predicted scores. More standard classifiers can also be used by ranking the results using the probability of a question belonging to a given class. For example, in our case, we can train a Logistic model or any other binary classifier $\mathcal{L}(f)$ and then label the question with the highest likelihood of being selected from a given list of available questions:

$$D(Q', F) = \underset{q \in Q', f \in F}{\operatorname{argmax}} \; P(q|\mathcal{L}(f)) \tag{7}$$

Besides the pointwise approach, LTR algorithms can also use *pairwise* and *listwise* algorithms [13]. On the one hand, the pairwise method uses binary classifiers for comparing question pairs within a list in order to decide the most relevant one. Such an approach has the advantage of taking into account the relation between the questions that need to be ranked. On the other hand, the listwise approach uses machine learning methods that try to optimise a general evaluation measure such as the Mean Reciprocal Rank (MRR). The advantage of such a method is that it can be optimised on a particular evaluation metric. In our case, such a metric can be Precision at one ($P@1$) since we want to identify the one question selected by answerers at a time. Such question should be ranked on top for accurately predicting questions selections.

In this paper we evaluate *Random Forests* as a pointwise method. LambdaRank [16] as a pairwise approach, and, ListNet [4] for a listwise method. These algorithms are implemented on top of the RankLib[8] and Weka[9] frameworks.

4. DATASET

Our work and experiment are performed on the Stack Exchange *Cooking* (CO) Q&A community, which supports a range of Q&A features such as *voting*, *reputation* and *best answer labelling*. Our data is extracted from the April 2011 Stack Exchange public dataset [10] and consists of $3,065$ questions, $9,820$ answers and $4,941$ users. In our experiment we randomly selected 100 users out of the 283 users that have answered at least five questions. Users with less activity are currently deemed too underrepresented for our prediction task.

In our dataset, on average the number of questions available for each user to select and answer at every observed selection time is 328 questions. There were 1757 question-selections made by our 100 users, which translates to over 575K question-selection possibilities to consider, and many millions of feature calculations.

Note that in some Q&A platforms, such as the one we experiment with, users are allowed to answer questions multiple times or answer questions that have already been marked as resolved. In our datasets, we did not find cases of multiple answers by the same user for the same question. However, we found that in many cases, users were answering after a question was marked as solved (71%). The reason for this behaviour might be due to two reasons:

1. User may feel that they can provide an answer that is better than the existing best answer, or;

2. Although the best answer has already been posted, the question may not yet be marked as resolved.

To resolve this issue, we include all recently resolved questions (i.e. less than three weeks old) to the list of available questions if a user has not already selected them. Such an approach enables us to reduce the number of questions affected by the previous issue to only 5%. We discard these remaining questions from our dataset in order to focus on user activities that highlight a more common behaviour (i.e. users that do not reply to questions that have been resolved a long time ago).

5. EXPERIMENT & EVALUATION

In this section we apply LTR models for identifying questions that users are most likely to answer, using the dataset described in the previous section. In order to validate our answering behaviour model, we try to predict which questions a user is going to select and answer.

We train a model for each user, using a 80%/20% chronologically ordered training/testing split based on the number of answers posted by each user. Then, for each user, we generate their historical question-selection lists and attach the *user*, *question* and *thread* features. Then, in each list we label selected questions as 1 and unselected questions as 0. We merge the training/testing lists and train LTR models, excluding any information on user selections from the testing set.

Three different types of LTR models are trained in this experiment (Section 3.4):

1. A pointwise algorithm based on *Random Forests*.

2. A pairwise *LambdaRank* model.

3. *ListNet*, a listwise algorithm.

For baselines, in addition to these algorithms, we also use *question age* (selecting most recent question, as in [12]) and *topic affinity* (selecting the question that is most similar to user topics, using Formula 2 in Section 3.3), as well as a *random* algorithm that selects a question randomly.

We evaluate each model using the Mean Reciprocal Rank (MRR) and the Mean Average Precision at n ($MAP@n$) metrics. For the $MAP@n$ metric we compare the results at different levels using $n = \{1, 2, 4, 8, 16, 32\}$. MRR represents the average rank of the relevant question in each list, and $MAP@n$ can be seen as the average position of the relevant question within the top n items of each list.

We run two experiments:

[8] `http://sourceforge.net/p/lemur/wiki/RankLib`.
[9] `http://www.cs.waikato.ac.nz/ml/weka`.
[10] `www.clearbits.net/get/1698-apr-2011.torrent`

1. First experiment compares the performance of our models for identifying selected questions for 100 users, using *user*, *question* and *thread* features separately, then all features combined. We also evaluate the models using the baseline features described above.

2. The second experiment focuses on evaluating the influence of each feature on question selection behaviour in order to better understand how users select the questions to answer. Using those results, we re-evaluate the previous models with a restricted number of features to look for any improvements in the results.

5.1 Results: Model Comparison

For our first experiment, we train our models on different feature subsets and compare the results using the previous metrics in order to better understand the importance of each features groups (Table 1). As expected, the *Random* approach performs very poorly with $MRR = 0.007$. As for *topic affinity*, it also proved incapable of making any accurate predictions of answerer behaviour. The *question age* model performs better, with $MRR = 0.094$ and the highest $P@1$ of 0.036. Hence a ranking solely based on the age of questions can enable users to find the question they are willing to answer within the first 10 questions. Much higher $P@1$ can be expected in communities where the default is to organise questions by recency, such as Yahoo! Answers, where a $P@1$ of 0.2445 was reported in [12]. We also observe that *user features* when used alone perform the worst overall, whereas *question features* provide a better average ($MMR = 0.182$) across all models.

Although *ListNet* performs better than the other approaches and baselines with $MMR = 0.139$, user features alone did not provide a good prediction of selected questions. This could be because most *user* features are only useful when linked with *question* and *thread* features. For example, the *reputation* of a user may be only used while coordinated with the average reputation of a thread.

In our prediction models, except for *ListNet*, *Question features* shows more value than *user features* with an average ($MMR = 0.182$) across all models. The best performing model is *Random Forests* with $MRR = 0.397$ which is much higher than both *LambdaRank* (+88.92%) and *ListNet* (+73.30%). *Thread features* seem less useful than *question features* with an average $MRR = 0.135$

When combining all features, *Random Forests* provides the best results with $MMR = 0.446$ meaning that selected questions are found on average in the 2^{nd} or 3^{rd} position. We also get $P@1 = 0.307$, a gain of +88.27% over our best baseline. Combining all features enables the ranking method to better consider the relations and influences between users, threads and questions, which seem to improve our predictions.

In general we observe that *Random Forests*, a pointwise algorithm, performs much better than the other models. Although in theory listwise methods are expected to perform better [13], in our case the simplest form of LTR models generated the best results. Such results may be explained by the fact that contrary to both *LambdaRank* and *ListNet*, the pointwise *Random Forests* method was optimised specifically for identifying question selection behaviour (i.e. there is only one relevant document per list).

5.2 Results: Feature Assessment

In this section, we aim to identify the features that are most useful for predicting question selection behaviour. We use standard Correlation Feature Selection (CFS) and Information Gain Ratio (IGR) by converting our problem into a classical binary classification task since such methods are not designed for LTR tasks. For each method we use 10-fold cross validation on the training dataset.

We also propose to rank features by dropping each individual feature one by one based on the full *Random Forests* LTR model and accounting for the drop in MRR (ablation method). Such a method has the advantage of including the LTR structure of our approach even though it does not take into account correlations between each behaviour feature. We apply the feature drop approach on the training split and evaluate it on the testing set defined earlier in Section 5. Finally, we merge the different ranks obtained from previous methods by calculating the average rank of each feature using CFS, IGR and feature-drop ranks.

The top rankings obtained by each individual method are listed in Table 2. The top fifteen features obtained from each feature selection method contain measures from each of the groups discussed in Section 3.3. Most of the top features are *question* features (40%) followed by the *thread* (31%) and *user* (29%) groups. Such results seem to indicate that all predictor types are equally important in determining the behaviour of answerers. This result is not really surprising as the best model obtained so far is generated when all the features are used.

Although IGR and CFS share similar top fifteen features (73%), it appears that the feature drop methodology produce quite different results by sharing only 20% of the features with IGR and 27% with CFS. Such a result is very likely due to the difference in methodology used by each feature selection method. Both IGR and CFS are applied on a simplification of our LTR tasks (i.e. binary classification task) while feature drop method is applied on the full *Random Forest* LTR model.

Table 3 compares the average ranks of each of our features. Most of the top fifteen features are *question* features. The top two *question* features are the number of hyperlinks contained in questions (*referral count*) and the reputation of the previous asker's questions (*questions reputation*). Other important features are *question age* and the total *number of answers* received by the question asker. Other *question* features show that users' choices of questions to answer are largely affected by the popularity of the previous questions of a given asker (*number of answers* and *questions reputation*). Looking in more detail into the first two features (Figure 2) we observe that users are more likely to select questions with a low referral count ($p = 1.53 \times 10^{-31}$)[11] and questions from users that have a high *question reputation* ($p = 1.20 \times 10^{-37}$). Such results suggest that users prefer to answer questions from popular askers and questions that do not require reading additional (hyperlinked) material.

By inspecting the distribution of the *LIX* metric we see that questions with simpler answers draw more selections

[11]We consider that the null hypothesis (H_0) is given when *there is no impact of referral count on question selection behaviour* and perform a two tailed t-test for understanding if *high or low referral count is associated with question selections* (H_1). We calculate the other p-values of other features similarly.

Table 1: Mean Reciprocal Rank (MRR) and Mean Average Precision ($MAP@n$) for identifying the most likely question-selection for 100 users randomly selected from those with more than 5 question answers for *Cooking*.

			Mean Average Precision ($MAP@n$)					
Model	Feature	MRR	$MAP@1$	$MAP@2$	$MAP@4$	$MAP@8$	$MAP@16$	$MAP@32$
Baseline	Random	0.007	0.007	0.007	0.007	0.007	0.007	0.006
	Question Age	0.094	0.036	0.053	0.069	0.082	0.089	0.090
	Topic Affinity	0.018	0.000	0.004	0.007	0.008	0.009	0.011
Random Forests	Observer	0.048	0.023	0.031	0.036	0.039	0.041	0.042
	Question	0.397	0.279	0.350	0.384	0.391	0.393	0.394
	Thread	0.246	0.212	0.222	0.224	0.225	0.234	0.239
	All	0.446	0.307	0.380	0.428	0.440	0.444	0.444
LambdaRank	Observer	0.045	0.018	0.018	0.023	0.028	0.033	0.038
	Question	0.044	0.028	0.032	0.034	0.036	0.037	0.039
	Thread	0.046	0.000	0.000	0.001	0.001	0.029	0.043
	All	0.234	0.222	0.222	0.223	0.226	0.227	0.228
ListNet	Observer	0.139	0.059	0.089	0.110	0.121	0.126	0.133
	Question	0.106	0.041	0.053	0.074	0.085	0.095	0.099
	Thread	0.112	0.039	0.056	0.076	0.094	0.098	0.104
	All	0.114	0.036	0.066	0.081	0.093	0.101	0.107

($p < 2.2 \times 10^{-16}$). Similarly, answerers seem to prefer questions that have not attracted skilled answerers yet ($p < 2.2 \times 10^{-16}$). Finally, the top *user* features are all related to user ability to answer and ask questions. Such features are probably highly ranked as they can be used for differentiating knowledgeable users from less skilled answerers. Therefore, a ranking model can use *user* information to adapt the ranking depending on the type of potential answerer.

5.2.1 Feature Reduction

We trained the *Random Forests* model for each set of features selected by the CFS, IGR, features drop, and average ranking methods. In order to select the minimal and most effective number of features, we gradually add in features according to their discriminative power and determine the best number of features based on the MRR score (Figure 3).

As shown in Figure 3, for each ranking approach, the optimum number of features are: 1) IGR: 60 ($MRR = 0.458$); 2) CFS: 58 ($MRR = 0.451$); 3) Feature drop (ablation method): 58 ($MRR = 0.491$), and 4) Average rank: 52 ($MRR = 0.463$). This implies that almost all our features are required for training the *Random Forests* LTR model. Nevertheless, we observe that by using the feature drop subset, we can outperform the best previous result by more than 9% MRR (Table 4). Although it appears that feature selection does not reduce significantly the number of features required for getting quality behaviour predictions, we can observe that by using only the top 15 features from average ranking, we get a result similar to our best previous result. As a consequence, it is possible to reduce the computational complexity of the behaviour model without sacrificing much prediction ability.

Table 3: Top fifteen features ranked using their average rank computed by merging the Information Gain Ratio, Correlation Feature Selection and Features Drop results for *Cooking*. Type of feature is indicated by U/Q/T for User/Question/Thread.

	Merged Rank (IGR+CFS+Drop)	
R.	AR.	Feature
1	1.66	Referral Count (Q)
2	6.66	Question Reputation (Q)
3	9.33	Number of Answers (Q)
4	10.33	Average Readability LIX (T)
5	10.33	Average Answer Success (T)
6	11.66	Answer Success Ratio (U)
7	14.66	Question Age (Q)
8	17.33	Number of Questions (U)
9	18.33	Questions Success (U)
10	18.33	Reputation (Q)
11	18.66	Average Answer Success Ratio (T)
12	18.66	Polarity (Q)
13	21	Question Success (U)
14	21.33	Number of Answers (U)
15	21.33	Average Readability Fog (T)

6. DISCUSSION

Modelling the behaviour of answerers is a complex task and it is generally difficult to obtain high precision for such a problem [12]. In this paper, we showed that by using dynamic features and LTR algorithms, we obtained a good precision ($MAP@1 = 0.5168$). For the CO community, we found that

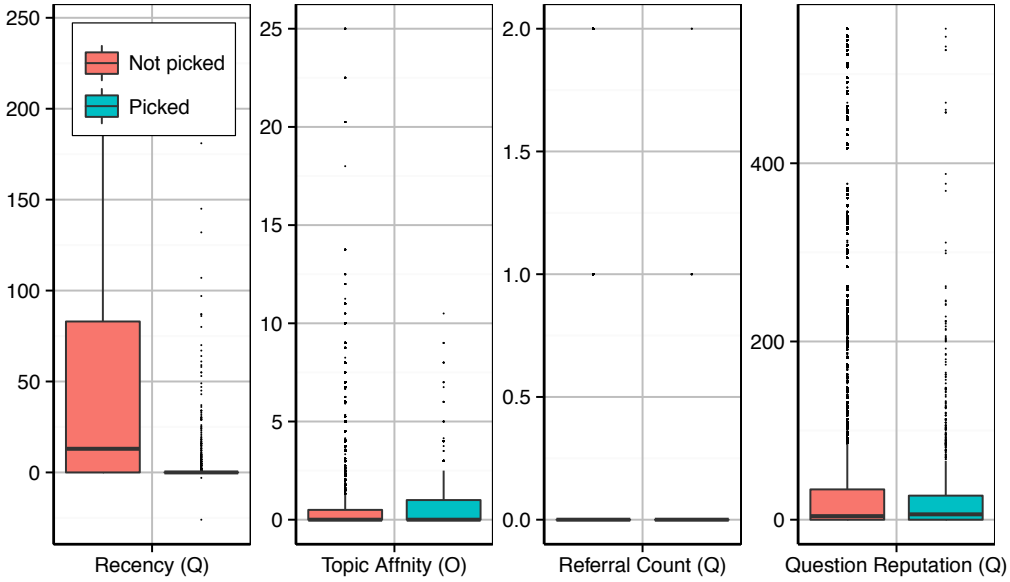


Figure 2: Box plots representing the distribution of different features and question selection for *Cooking*.

answering behaviour is mostly affected by three different features: the ability of askers to obtain good answers, the recency of questions and the syntactical complexity of existing answers. Questions asked by users that had difficulties to obtain good answers were more likely to get selected ($p = 9.723e-05$). Naturally, recent questions had a higher chance of being selected by answerers ($p < 2.2e-16$) and questions that had less complex answers were also more likely to be selected by answerers ($p < 2.2e-16$)

Our approach generated high MRR and $P@1$ ($MAP@1$) indicating that on average, the questions a user will select to answer are positioned within the first two list elements returned by our LTR model. Although our dataset consisted of a randomly selected 100 users only, our analyses considered all open and available questions as possible candidates for users to select to answer, which was 328 questions per user per time t. In [12] only the latest 20 questions were considered. To further optimise our approach, we plan to study whether the influence of our prediction features change when the size of the question list is smaller or bigger. Our analyses showed that using a smaller subset of features can produce high prediction accuracies, which facilitates applying our approach to much larger community datasets.

In order to better understand how answerers select their questions, we applied different automatic methods for ranking features according to their ability to model answering behaviour. Although such methods were originally designed for standard classification tasks (e.g. binary classification) and are theoretically suboptimal for LTR problems, we managed to maintain the quality of our results by cutting the number of required features considerably. As a consequence, we observed that for the CO community, it is not necessary to have many different predictors in order to identify the questions selected by users. Although it is necessary to validate this finding on bigger and more diverse communities, it is possible to apply it to the CO community in order to reduce the amount of computation required for creating behaviour models. In particular, by reducing the number of required features, it becomes possible to go beyond the 100 user sample that we used for our study of CO.

In this paper, we also removed 5% of the questions that were answered even if they were solved and older than three weeks. Although we argue that such a percentage of questions is unlikely to affect our results, future work could investigate the identification of these type of questions and study a method for predicting behaviour with those particular items.

Users' question selection time was known when predicting our answering behaviour. One extension is to also predict those selection times, as partially studied in [21] and [16]. This would enable us to predict when a user answers a question, as well as which question the user selected for answering.

Another point worth highlighting is that we experimented with a single Q&A community, on Cooking. It would be worth applying our approach to other communities on cooking, as well as to other communities on different topics, to see if similar behaviour patterns are exhibited. Such an extension is necessary to determine the portability of our findings across different communities and topics [18].

Finally, we limited our number of features to 62 *user*, *question* and *thread* features. Although there is potentially an infinite number of features that could be evaluated for the task presented in this paper, we found that question features were the most associated with question selection behaviour. Future work could investigate additional features in this particular area such as different readability metrics [8] or question complexity measures [1].

7. CONCLUSIONS

This paper proposed an approach for identifying the questions that users are most likely to answer in a Q/A website.

Table 2: Top fifteen features ranked using their average rank computed using Information Gain Ratio, Correlation Feature Selection and Features Drop for the *Cooking* dataset and the corresponding MRR performance. Type of feature is indicated by O/Q/T for Observer/Question/Thread.

	Info. Gain Ratio			CFS			Feature Drop.		
R.	AR.	IGR	Feature	AR	AC	Feature	R.	MRR	Feature
1	4	0.011	Answer Succ. Ratio (O)	1.8	0.022	Answer Succ. Ratio (O)	1	0.414	Referral Count (Q)
2	4.2	0.011	Referral Count (Q)	2.2	0.021	Referral Count (Q)	2	0.418	Nb. of Answers (Q)
3	6.8	0.010	Question Reputation (Q)	4.4	0.022	Question Age (Q)	3	0.418	Avg. Nb. of Questions (T)
4	7.7	0.010	Avg. Readability LIX (T)	5.7	0.022	Question Reputation (Q)	4	0.422	Asker Reputation (Q)
5	8.1	0.010	Question Age (Q)	6.4	0.023	Avg. Readability Fog (T)	5	0.423	Avg. Question Reputation (T)
6	9.1	0.013	Avg. Reputation (O)	7.5	0.022	Nb. of Answers (Q)	6	0.425	Avg. Question Reputation (O)
7	11.4	0.010	Answers Succ. Ratio (Q)	9.2	0.022	Avg. Readability LIX (T)	7	0.427	Topic Reputation (O)
8	11.8	0.010	Topic Affinity (Q)	9.5	0.022	Avg. Referral Count (T)	8	0.427	Topic Affinity (O)
9	11.9	0.010	Avg. Referral Count (T)	13.9	0.022	Topic Affinity (Q)	9	0.430	Readability LIX (Q)
10	12	0.346	Avg. Quest. Reputation (Q)	14	0.022	Avg. Quest. Succ. Ratio (T)	10	0.431	Polarity (Q)
11	12.6	0.010	Avg. Quest. Succ. Ratio (T)	15.1	0.021	Avg. Reputation (O)	11	0.432	Length (Q)
12	14	0.010	Avg. Polarity (T)	15.2	0.022	Avg. Answer Succ. (T)	12	0.432	Avg. Reputation (T)
13	14.2	0.010	Avg. Answers Succ. Ratio (T)	15.6	0.022	Questions Succ. (O)	13	0.434	Question Reputation (Q)
14	14.4	0.010	Avg. Answer Succ. (T)	16.3	0.022	Nb. of Posts (O)	14	0.434	Avg. Reputation (O)
15	15.4	0.009	Questions Succ. (O)	17.2	0.022	Nb. of Answers (O)	15	0.434	Nb. of Questions (O)

Table 4: Mean Reciprocal Rank (MRR) and Mean Average Precision ($MAP@n$) for identifying the most likely question-selection for 100 users randomly selected from those with more than 5 answers for *Cooking* using different feature selections approaches.

			Mean Average Precision ($MAP@n$)					
Model	Feature	MRR	$MAP@1$	$MAP@2$	$MAP@4$	$MAP@8$	$MAP@16$	$MAP@32$
	All	0.446	0.307	0.380	0.428	0.440	0.444	0.444
	IGR	0.329	0.326	0.399	0.438	0.453	0.456	0.456
	CFS	0.451	0.326	0.394	0.433	0.447	0.449	0.449
	Drop	0.491	0.364	0.439	0.478	0.485	0.488	0.489
	IGR+CFS+Drop	0.463	0.326	0.398	0.449	0.459	0.461	0.462
LambdaRank	All	0.234	0.222	0.222	0.223	0.226	0.227	0.228
	IGR	0.227	0.209	0.213	0.215	0.219	0.220	0.220
	CFS	0.226	0.212	0.213	0.215	0.219	0.220	0.220
	Drop	0.245	0.233	0.233	0.233	0.235	0.236	0.239
	IGR+CFS+Drop	0.228	0.212	0.216	0.217	0.220	0.221	0.221
ListNet	All	0.114	0.036	0.066	0.081	0.093	0.101	0.107
	IGR	0.105	0.039	0.062	0.073	0.083	0.091	0.098
	CFS	0.086	0.026	0.037	0.054	0.064	0.074	0.079
	Drop	0.111	0.041	0.053	0.070	0.088	0.097	0.106
	IGR+CFS+Drop	0.091	0.034	0.047	0.066	0.072	0.079	0.085

We introduced different LTR models for learning user behaviours, and showed that a *Random Forests* ranking model can identify question selections efficiently with a MRR of 0.491 with 52 features and an MRR of 0.441 with 15 features. We found that question selections are highly influenced by question features such as whether they contain hyperlinks, have already received some answers, or have been asked by a reputable user. We also found that users tend to answer more recent questions and that the readability of existing answers affect users' selections.

8. ACKNOWLEDGEMENT

This work is partly funded by the EC-FP7 project DecarboNet (grant number 265454).

9. REFERENCES

[1] G. Burel and Y. He. A question of complexity: Measuring the maturity of online enquiry communities. In *Proceedings of the 24th ACM Conference on Hypertext and Social Media*, HT '13, pages 1–10, New York, NY, USA, 2013. ACM.

[2] G. Burel, Y. He, and H. Alani. Automatic identification of best answers in online enquiry communities. *The Semantic Web: Research and Applications*, 2012.

[3] Y. Cao, H. Duan, C.-Y. Lin, Y. Yu, and H.-W. Hon. Recommending questions using the mdl-based tree cut model. In *Proceedings of the 17th international conference on World Wide Web*. ACM, 2008.

[4] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li. Learning to rank: From pairwise approach to listwise approach. In *Proceedings of the 24th International Conference on Machine Learning*, ICML '07, New York, NY, USA, 2007. ACM.

[5] S. Chang and A. Pal. Routing questions for collaborative answering in community question answering. In *Proceedings of the 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining*. ACM, 2013.

[6] G. Dror, Y. Koren, Y. Maarek, and I. Szpektor. I want to answer; who has a question?: Yahoo! answers recommender system. In *Proceedings of the 17th ACM*

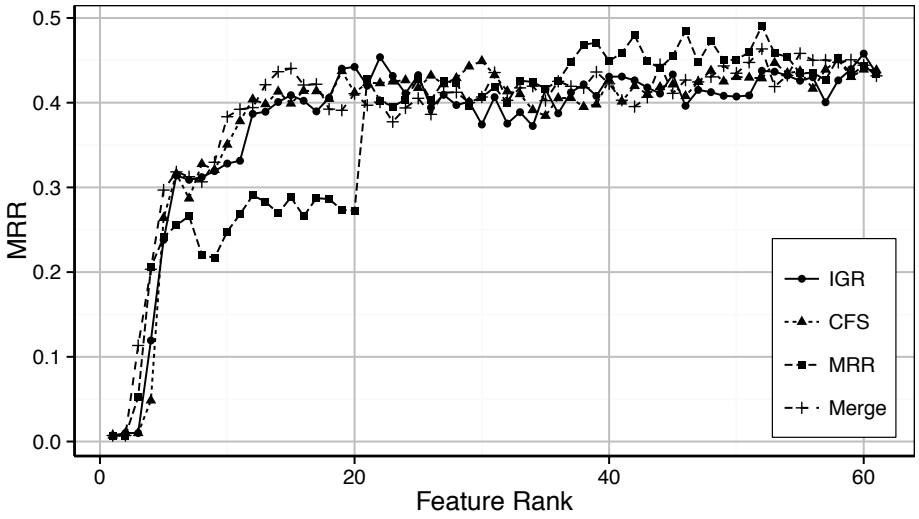


Figure 3: *MRR* vs. feature rank for the Information Gain Ratio, Correlation Feature Selection, Features Drop feature selection and Merged Ranking methods for *Cooking*.

SIGKDD international conference on Knowledge discovery and data mining. ACM, 2011.

[7] H. Duan, Y. Cao, C.-Y. Lin, and Y. Yu. Searching questions by identifying question topic and question focus. In *ACL*, 2008.

[8] L. Feng, M. Jansche, M. Huenerfauth, and N. Elhadad. A comparison of features for automatic readability assessment. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 276–284, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[9] L.-C. Lai and H.-Y. Kao. Question routing by modeling user expertise and activity in cQA services. In *The 26th Annual Conference of the Japanese Society for Artificial Intelligence*, 2012.

[10] B. Li and I. King. Routing questions to appropriate answerers in community question answering services. In *Proceedings of the 19th ACM international conference on Information and knowledge management*. ACM, 2010.

[11] B. Li, I. King, and M. R. Lyu. Question routing in community question answering: putting category in its place. In *Proceedings of the 20th ACM international conference on Information and knowledge management*. ACM, 2011.

[12] Q. Liu and E. Agichtein. Modeling answerer behavior in collaborative question answering systems. In *Advances in Information Retrieval*. Springer, 2011.

[13] T.-Y. Liu. Learning to rank for information retrieval. *Found. Trends Inf. Retr.*, 3(3), Mar. 2009.

[14] M. J. Pazzani and D. Billsus. Content-based recommendation systems. In *The adaptive web*. Springer, 2007.

[15] M. Qu, G. Qiu, X. He, C. Zhang, H. Wu, J. Bu, and C. Chen. Probabilistic question recommendation for question answering communities. In *Proceedings of the 18th international conference on World wide web*. ACM, 2009.

[16] C. Quoc and V. Le. Learning to rank with nonsmooth cost functions. *NIPS'07*, 2007.

[17] F. Riahi, Z. Zolaktaf, M. Shafiei, and E. Milios. Finding expert users in community question answering. In *Proceedings of the 21st international conference companion on World Wide Web*. ACM, 2012.

[18] M. Rowe and H. Alani. Mining and comparing engagement dynamics across multiple social media platforms. In *ACM 2014 Web Science Conference*, Bloomington, Indiana, USA, 2014.

[19] M. Surdeanu, M. Ciaramita, and H. Zaragoza. Learning to rank answers to non-factoid questions from web collections. *Computational Linguistics*, 37(2), 2011.

[20] I. Szpektor, Y. Maarek, and D. Pelleg. When relevance is not enough: promoting diversity and freshnessin personalized question recommendation. In *Proceedings of the 22nd international conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2013.

[21] Y. Tian, P. S. Kochhar, E.-P. Lim, F. Zhu, and D. Lo. Predicting best answerers for new questions: An approach leveraging topic modeling and collaborative voting. In *Social Informatics*. Springer, 2014.

[22] H. Wu, Y. Wang, and X. Cheng. Incremental probabilistic latent semantic analysis for automatic question recommendation. In *Proceedings of the 2008 ACM conference on Recommender systems*. ACM, 2008.

[23] Y. Zhou, G. Cong, B. Cui, C. S. Jensen, and J. Yao. Routing questions to the right users in online communities. In *Data Engineering, 2009. ICDE'09. IEEE 25th International Conference on*. IEEE, 2009.