Information Sciences 608 (2022) 696–717

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

Feature-based question routing in community question answering platforms

Soroosh Sorkhani [a,*], Roohollah Etemadi [a], Amin Bigdeli [a], Morteza Zihayat [a], Ebrahim Bagheri [a]

[a] *Ryerson University, Canada*

ARTICLE INFO

Article history:
Received 26 December 2021
Received in revised form 17 June 2022
Accepted 19 June 2022
Available online 27 June 2022

Keywords:
Question routing
Expert finding
Embedding
Learning to rank

ABSTRACT

Community question answering (CQA) platforms are receiving increased attention and are becoming an indispensable source of information in different domains ranging from board games to physics. The success of these platforms dependent on how efficiently new questions are assigned to community experts, known ascalled *question routing*. In this paper, we address the problem of question routing by adopting a learning to rank approach over five CQA websites in the context of which we introduce 74 features and systematically classify them into *content-based* and *social-based* categories. Our extensive experiments on datasets from five real online question answering websites indicate that content-based features related to *tags* and *topics* as well as social features that are related to *user characteristics* and *user temporality* are effective for question routing. Our work shows the ability to improve performance compared to the state-of-the-art neural matchmaking methods that lack the interpretability offered by our work. The improvement can be as high as on average 2.47% and 1.10% in terms of common ranking metrics, Normalized Discounted Cumulative Gain (NDCG) and Mean Average Precision (MAP) respectively, compared to our best baselines.

1. Introduction

Community Question–Answering (CQA) platforms, such as *Stack Overflow*,[1] and *Quora*[2] have become tools that are used on a daily basis by many users. Such platforms utilize human-generated answers in order to respond to questions raised by the users. Given a question posted by a user, other users can provide and submit their answers. Then, one of the submitted answers may be selected as an *'accepted answer'* by the user who posted the question. Users can also provide feedback in the form of up-votes or down-votes for the questions and/or answers. Fig. 1 displays a question and its answers on Stack Overflow. With the growing popularity comes challenges associated with the efficiency and quality of the provided answers [1]. Therefore, identifying the best user to answer a question is of paramount importance. At the time of writing this paper, there are near *3 million* unanswered questions and more than *7 million* questions without an accepted answer on Stack Overflow.[3] The recommendation

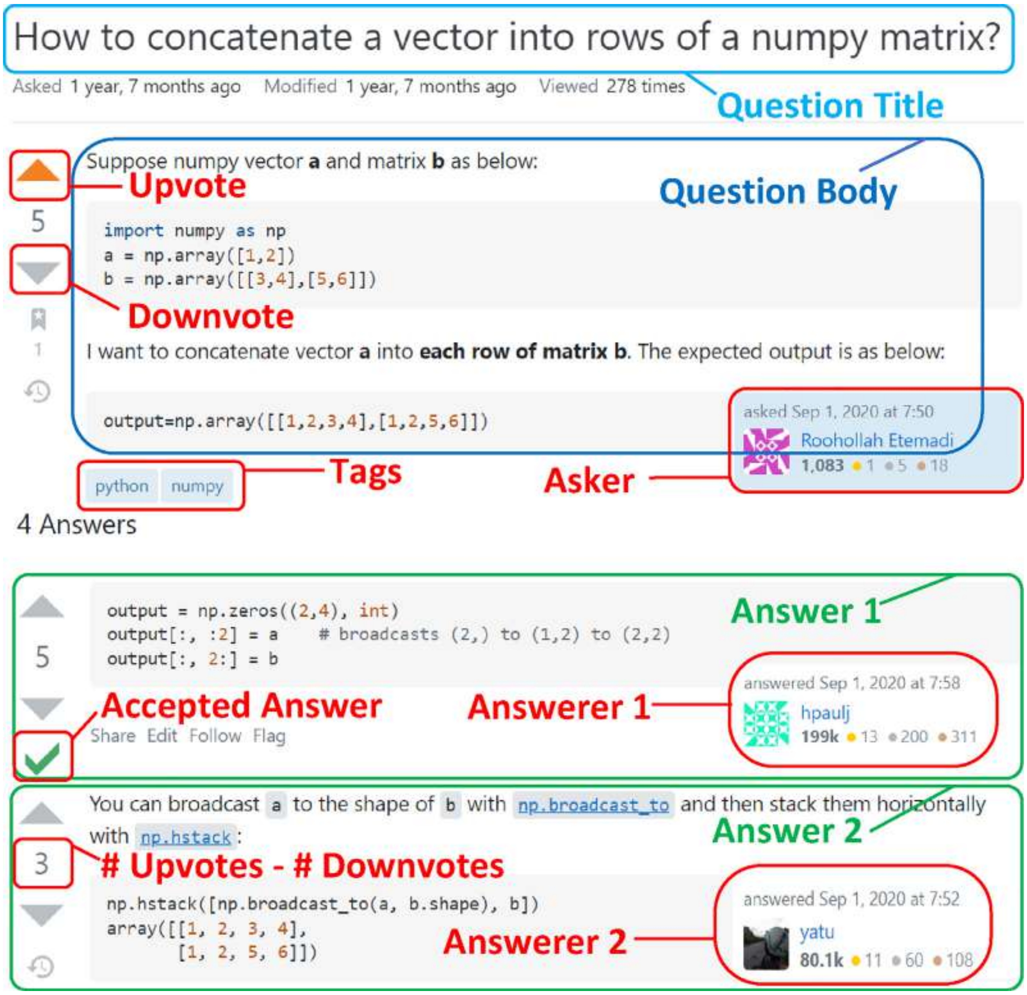
* Corresponding author.
E-mail addresses: soroosh.sorkhani@ryerson.ca (S. Sorkhani), etemadir@ryerson.ca (R. Etemadi), abigdeli@ryerson.ca (A. Bigdeli), mzihayat@ryerson.ca (M. Zihayat), bagheri@ryerson.ca (E. Bagheri).

[1] https://stackoverflow.com/.

[2] https://www.quora.com/.

[3] https://data.stackexchange.com/.

https://doi.org/10.1016/j.ins.2022.06.072
0020-0255/© 2022 Elsevier Inc. All rights reserved.



How to concatenate a vector into rows of a numpy matrix?

Asked 1 year, 7 months ago Modified 1 year, 7 months ago Viewed 278 times

Suppose numpy vector **a** and matrix **b** as below:

```
import numpy as np
a = np.array([1,2])
b = np.array([[3,4],[5,6]])
```

I want to concatenate vector **a** into **each row of matrix b**. The expected output is as below:

```
output=np.array([[1,2,3,4],[1,2,5,6]])
```

asked Sep 1, 2020 at 7:50
Roohollah Etemadi
1,083 ● 1 ● 5 ● 18

python numpy

4 Answers

```
output = np.zeros((2,4), int)
output[:, :2] = a    # broadcasts (2,) to (1,2) to (2,2)
output[:, 2:] = b
```

answered Sep 1, 2020 at 7:58
hpaulj
199k ● 13 ● 200 ● 311

You can broadcast `a` to the shape of `b` with `np.broadcast_to` and then stack them horizontally with `np.hstack`:

```
np.hstack([np.broadcast_to(a, b.shape), b])
array([[1, 2, 3, 4],
       [1, 2, 5, 6]])
```

answered Sep 1, 2020 at 7:52
yatu
80.1k ● 11 ● 60 ● 108

Fig. 1. A question and answer on Stack Overflow along with the upvotes and downvotes, tags, and other elements.

of the right question to the best expert would facilitate the resolution of such pending questions and would make question answering platforms more vibrant.

The literature has already explored ways through which experts and questions can be effectively paired, a process which is referred to as *question routing* in the community [2,3]. The main idea is to route newly posted questions to those experts that have sufficient background knowledge and are most likely to answer them. Given the importance of the problem, recent works have developed a diverse range oftechniques to perform *question routing* [4]. One of the popular approaches, similar to the work in this paper, is to leverage a feature-based method to measure the relevance between experts and the available questions [5]. The benefit of such approaches is that they offer a deeper understanding of the characteristics that lead to an effective question routing strategy. On the other hand, a large number of works in the past few years have been on the introduction of end-to-end deep neural network architectures that can find relevance between users and questions [4]. While these approaches have shown strong performance, they lack transparency and interpretability when providing their recommendations. It is possible to interpret and explain the recommendation of these deep neural techniques based on Post hoc (post-model) interpretations, which are primarily based on the explanation of individual decisions or a general interpretation of the complete model [6]. In the latter case, post hoc methods build a second model to explain theexisting one while in the former case, they provide explanations for individual predictions by identifying the contributions of each feature in the input considering a particular model prediction outcome. It has been shown that the outcomes might be appealing if audiences are knowledgeable machine learning or artificial intelligence researchers, as they leverage the statistical analysis of the feature importance distributions to debug the models. However, such outcomes may be less useful for domain-specific audiences. This type of interpretation is valuable, for instance, in cases when the user would want to be informed why a given question was recommended to them. However, they would fall short of providing actionable insights into the motivating factors that would persuade and convince users to engage with a new question. For the sake of understanding the dynamics of a community question answering platform, adopting a feature-based approach with comparable performance to neural network techniques would be desirable, which would provide insight into what are the factors that lead to effective question routing.

As such, the main objectives of this work are multifold and outlined as follows:

O1: Building on existing work in feature-based question routing by extensively introducing and systematically classifying a comprehensive set of features for question routing.
O2: Performing feature ablation studies to understand the impact and importance of features for the question routing task.
O3: Training a learning to rank model based on the introduced features that provides competitive (superior) performance to the state-of-the-art neural ranking techniques.

In our work, we define two broad classes of features, namely those that refer to the content of the posted questions (i.e., *content features*) and those that relate to other social aspects of the platform (i.e., *social features*). The class ofcontent features covers important aspects such as tags, content similarity, readability and topics, while the class ofSocial features addresses user characteristics, temporal aspects, and network characteristics.

The major distinguishing contributions of our work can be enumerated as follows:

(1) We collected and developed 74 features from user (expert) and question characteristics, as well as their relationships. These represent the social and textual information for question routing. To the best of our knowledge, this study is the first to consider a comprehensive set of features for question routing. In this study, 37 out of 74 features are proposed for the first time in the context of CQA.
(2) Based on the proposed features, we developed a novel learning to rank model with explainable results to address the question routing problem. This is the first study that utilizes the power of learning to rank models to rank users with respect to a given question in community question answering platforms.
(3) We studied the impact of each feature using extensive experiments on five real online platforms' datasets from a variety of domains and datasetsizes. This sheds light on the performance of our proposed features and the state-of-the-art data-driven features used in the context of question routing.
(4) We adopted the Gini score, as a common metric to study the degree of importance of each feature in our model, in orderto enhance interpretability and provide actionable insights into question routing. We also discuss the most and least influential features in the ranking model on each dataset.

The rest of this paper is organized as follows. Section 2 summarizes the existing techniques related to our research. The formal definition of the problem is presented in Section 3. The conceptual overview of our proposed approach and feature representation are presented in Section 4. Section 5 covers our experiments and findings. The practical implications of this work are presented in Section 6. Finally, Section 7 concludes the paper.

2. Related work

Collaborative and community question answering (CQA) platforms are now an essential source of knowledge and information on the Web. The success of such platforms relies, in part, on the efficient routing of questions to appropriate and willing experts, often known as *question routing*. In their seminal work, [15] proposed three main steps to build a question routing framework: (a) creating a question profile, (b) creating a user profile, and (c) matching the profile of a given question to the user profiles. We adopt this classification framework to summarize existing work in the literature in Table 1.

Several researchers have viewed the problem of question routing as one that can be addressed through *language models*. For instance, the work by [7] performs question routing by building question and user profiles based on a bag of words representation and matching the profiles based on a language model. Later, other authors [8,16] improved the performance of question routing by considering additional features such as time-based features for building question and user profiles and benefited from the query likelihood language model for matching between questions and users. Categories were later added to the considered features [16] when constructing user and question profiles. Several studies, such as [9,10], represented user and question profiles with topic model vectors based on which matching between users and questions are accomplished through the similarity of their topic representations. Sahu et al. [17] employed only tag information for performing topic modeling over users' interests. They also consider community feedback to compute user expertise.

A different class of methods views the problem of question routing as one of classification, ranking and/or a missing value problem in order to utilize more diverse features to represent question and user profiles. For example, Luo et al. views the question routing task as a classification problem and extracts features from IBM Connections (an enterprise CQA platform) and non-CQA sources to capture features such as employees' (users) expertise, willingness (e.g. amount of previous activity) and readiness (e.g. current load) [12].

In contrast to such methods, recently, several neural ranking methods have been proposed [18–20,4,21] that can be applicable to the question routing task. For example, Mitra et al. [20] have proposed a neural architecture for performing matchmaking using local and learned distributed representations. Dai et al. [18] have proposed a neural approach that matches

Table 1
Summary of most relevant related work.

Category	Citation	Question Profile	User Profile	Matching Model	Ground Truth	Eval Metric	Dataset	Features			
								Textual	Metadata	Temporal	Network
Language Model based	[7]	BoW	BoW	QLLM, RM, Cluster-based LM	Actual Answerers	MRR	Wondir	✓	×	×	×
	[8]	BoW	BoW + Answer quality + Availability	QLLM + Jelinek-Mercer smoothing	Actual Answerers	MRR	Yahoo! Answers	✓	✓	✓	×
Topic Model based	[9]	BoW	BoW, LDA, STM	Cosine similarity, QLLM, topic similarity	Best Answerer	S@n	Stack Overflow	✓	×	×	×
	[10]	BoW + LDA topic model + Category information	BoW/LDA + Category + Temporal	Cosine similarity + Diversification	Online experiment (A/B test)	Activity level	Yahoo! Answers	✓	✓	✓	×
Classification and Ranking based	[11]	Textual, QU relationship	Activity, Expertise	Classification + Ranking; SVM + SVM Rank (pairwise)	Actual Answerers	P@n, MAP, MRR	Stack Overflow	✓	✓	×	×
	[12]	Q type, Topic - BoW	Expertise, Motivation, Availability	Classification; Linear regression + Diversification	Actual Answerers	Accuracy	IBM Connections	✓	✓	✓	×
	[4]	Graph embedding	Graph embedding + Textual	Ranking; Regression	Best answerer	P@n, Accuracy	8 Stack Exchange sites	✓	✓	×	✓
Collaborative based	[13]	Textual, topic - LDA, LSA	Activity, Expertise, Non-QA	Clustering; k-nn	Actual community	P@n, MRR	IBM Connections	✓	✓	✓	✓
	[14]	Temporal, topic - LDA, keywords	Expertise, Availability, Compatibility	Greedy algorithm	Actual community	P@n, R@n, MSC	Stack Overflow	✓	✓	✓	✓

questions and answers by representing n-grams of various lengths in an embedding space rather than the exact matching of questions and answers. Similarly, Yang et al. [21] have introduced an attention-based neural matching model for question routing. They haveused a value-shared weighting schema to aggregate matching signals. Despite their superior effectiveness, these methods do not have the interpretability of the feature-based methods.

Finally, there have also been works in the literature that extend the idea of question routing to one of finding collaborative groups of experts that can answer questions collaboratively. The idea of these methods is that given the interdisciplinary nature of questions, it is likely that each question would be efficiently answered if more than one expert was involved in the process of answering that question. Such works attempt to find those users who have synergistic expertise and are willing to answer questions collaboratively. Pal et al. [13] proposed a k-nn clustering based method to find groups of experts. They later employed a greedy algorithm to find collaborative teams [14].

2.1. Learning to rank in CQA

The Learning to Rank (LTR) (IR) task is defined as a process for training a model to automatically rank new objects according to their relevance, preference or importance [22]. Learning to rank methods have been utilized in different areas including but not limited to document retrieval, email routing, product rating, anti-web spam, and problems in the question answering domain. Table 2 summarizes the research studies in which LTR approaches have been used to address different research questions in the context of community question answering. LTR models are common tools for ranking answers in CQA. In [23], the authors offered 8 groups of features consisting of 186 individual features including contextual and user characteristics features to rank answers and predict their quality before receiving community feedback. They investigated a comprehensive set of features with different combinations to identify influential features contributing to the quality of answers. The authors showed that their method was also able to handle user cold start (users who are contributing for the first time in the CQA systems). A pair-wise random forest learning to rank algorithm was employed in their study. In another study, Ji et al. [11] introduced a total of 8 features within three broad feature categories, namely *question-specific statistical features*, *user-specific statistical features*, and *question-user relationship features* to rank answers in CQA. They compared topic-based and language model-based methods based on these features and showed that topic-based methods are more effective than language model-based methods. Burel et al. [24] employed LTR to recommend questions based on the user's past question-selection behavior. They utilized features such as user and question characteristics, temporal and readability features. However, their study lacked an investigation of topical, semantic similarity, and network features. Moreover, the authors in [25] trained a LTR model based on community feedback and similarity of a new question to the past questions a user has answered in order to address the problem of expert finding. In another study [26], textual features were defined in order to support ranking of questions in CQA and to retrieve similar questions. In more recent work, Jinhyuk et al. [27] used two separate RNNs (e.g., Bi-LSTM) on paragraphs of retrieved documents/answers and questions to rank paragraphs instead of ranking the answers with respect to a certain question in open-domain QA platforms. As illustrated in Table 2, to the best of our knowledge, many of the existing studies capture network-based, metadata-based and temporal features , which are also considered in our paper.

2.2. Contributions of this work

This work is the first study to consider a comprehensive set of features all together for question routing. Some features employed in this study are used in question routing research before, while some are used in studies with different purposes such as answer ranking [23], and expert finding [28]. To the best of our knowledge, 37 out of 74 features used in this paper are proposed for the first time in this context. The features used in this study are systematically classified into categories and

Table 2
Summary of papers that utilize learning to rank within the CQA domain.

Citation	Published in	Textual features	Metadata features	Temporal features	Network features	Dataset
hline [23]	SIGIR-2013	Style, structure and text similarity	Related to user	Very few and limited time-related features	None	Stack Overflow
[11]	CIKM-2013	Structure and text similarity	A few user-related features	None	None	Stack Overflow
[24]	ACMHT-2015	Structure and readability	User-related and tag features	The passed time since the question is posted	None	Stack Exchange Cooking
[25]	ICDMW-2015	Text similarity	User scores and tag features	None	None	Stack Overflow
[26]	COLING-2016	Text similarity	None	None	None	Qatar Living forum, Arabic medical forums
[27]	EMNLP-2018	Text similarity	None	None	None	CuratedTrec, WebQuestions, WikiMovies, SQuAD_OPEN

sub-categories. This gives the privilege of analyzing the influence and efficiency of each class on the performance of the question routing. Furthermore, we employed Gini importance to evaluate the effectiveness of each individual feature in the performance of our model. Approaches proposed in the literature lack the interpretability and transparency in the effect of the different feature classes on their performance.

Moreover, an LTR approach for the question routing problem is not proposed in any past studies. Leveraging the strengths of a learning to rank approach for ranking users according to a given question is an original perspective of this work. Additionally, the proposed approach along with the use of the extensive features has produced a ranking model that outperforms the state-of-the-art baselines for question routing.

3. Problem statement

We formulate the problem of question routing as follows. Let us assume that there are n questions denoted as $Q = \{q_1, q_2, \ldots, q_n\}$ in a CQA platform. Further, allow sets $A_i = \{a_1, a_2, \ldots, a_n\}, U_i = \{u_1, u_2, \ldots, u_n\}$ be answers and answerers of question q_i, respectively where u_k is the answerer of answer a_k. In addition, let $V_i = \{v_1, v_2, \ldots, v_n\}$ be the voting scores of answers of question q_i, where integer v_k is the difference between the up- and down-votes of answer a_k. Given a new question, the objective of the question routing task is to match a question to a ranked list of potential answerers. To this end, in this paper, our goal is to: (1) collect and introduce a set of features that can be extracted from data in a CQA platform for performing effective question routing; (2) build a learning-to-rank framework for question routing by utilizing the introduced features; and (3) investigate the impact of each feature in the performance of the question routing task.

4. The proposed approach

4.1. Overview

A high-level overview of our proposed approach is shown in Fig. 2. Given a set of questions and users in a CQA platform, we adopt an LTR approach to address the question routing task where all users on the CQA platform are considered to be potential answerers for a new question and areranked based on their relevance to the new question q_i. The objective our work is to effectively rank users based on their relevance to the question such that users at the top of the ranked list have a higher chance of providing an accepted answer for the newly posted question. To build the ranking model, different features representing users, questions and user-question relationships are proposed and are used to train an LTRmodel.

4.2. Feature representation

For a question q_i and ananswerer u_k, our question routing approach will predict the relevance of u_k to q_i using different sets of features representing both q_i, u_k and their interactions. Inspired by [11], we propose 74 features and categorize them into different groups from several perspectives. Each feature, either directly or indirectly, attempts to represent the relevance level of u_k to question q_i. As shown in Fig. 3, our features are classified into two broad categories, namely **content-based** and **social-based** features. Each feature is also categorized as being a feature that focuses on capturing a specific characteristic of the Question (Q), User (U), or the interaction between question-user (QU) pairs. Each set of features is formally defined in the following.

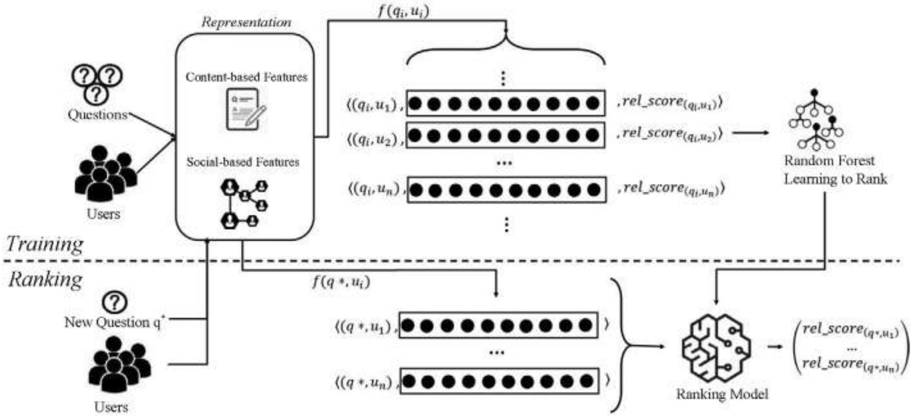


Fig. 2. Overview of our proposed approach. In the training phase, different features representing users, questions and user-question relationships are extracted and are used to train a learning to rank model. Then such a model is employed to rank existing users given a new question q^*.

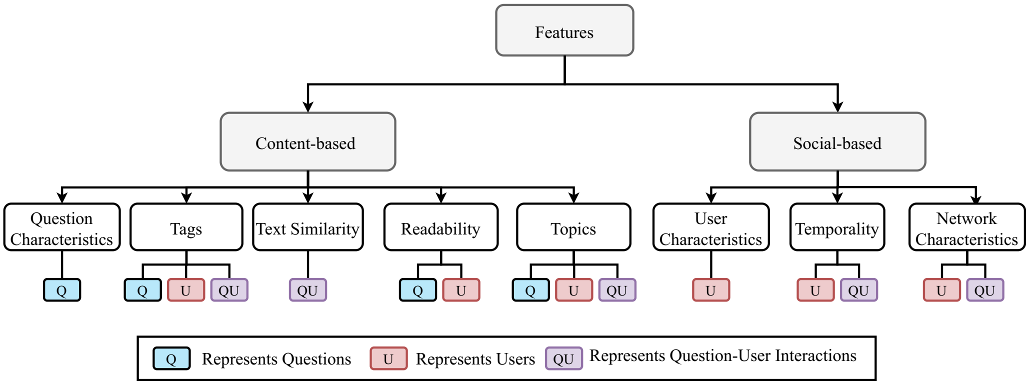


Fig. 3. The taxonomy of the proposed features. The features are categorized into two main groups, namely *content-based* or *social-based* features.

4.2.1. Content-based features

The content of answers and questions is a rich source of information to determine the relevance of a user to a new question. We carefully analyze such content and extract different features as follows. Given question q_i and its answerer u_k, Table 3 presents a summary of 46 features extracted from CQA content. The details of each feature sub-category are provided in the following subsections. Fig. 4.

Question Characteristics Features. Prior research in feature-based learning to rank techniques [29] has shown that there are certain objects (e.g., documents) that are preferred by the users. Therefore, it is reasonable to define features to represent the prior likelihood of a user's interests with regard to a certain question.

We define five features in the context of question routing, in order to capture such priors. In Table 3, the qc_1 feature indicates the presence of a code snippet in the question, since existing work suggests that a good-quality question would be one that contains code snippets [30]. Features qc_2 and qc_4 are the length (number of characters) of the text in the title and body of the question, respectively. Lastly, features qc_3 and qc_5 count the number of words in the title and body of the question. These features are aligned with the findings in [30] that has reported that title length is an effective indicator of a question's quality.

Tag Features. Tags are chosen by the user posting the question in order to indicate the domain or topic of the question. As such, questions with the same tags are often semantically similar to each other. It is necessary to consider tags given as Nie et al. [31] suggested that automatically generated topics are still not as practical as human-generated tags. Intuitively, when a user asks or answers a question with a specific tag, this can be considered to be an implicit sign of the user's interest or expertise. For example, if a user answers a question tagged with "#java", this could indicate that she is interested in, at least to some degree, questions related to "#java". We develop seven features that employ tag-related information to capture question-specific, user-specific and user-question interaction characteristics.

In addition to the number of tags and number of common tags, we define *tag specificity* (tg_6 in Table 3) to capture the consistency of question tags answered by the user [23]. The definition of specificity is adopted from [32], presented as *entropy*. The feature is computed as:

$$tg_6(u_i) = -\frac{1}{|Tg_{u_i}|}\sum_{j=1}^{|Tg_{u_i}|} P(t_j|u_i) ln(P(t_j|u_i)), \tag{1}$$

where Tg_{u_i} is the set of tags in the questions answered by user u_i and $P(t_j|u_i)$ denotes the probability of a tag t_j in the questions answered by u_i.

We also define *tag embedding similarity* (tg_7 in Table 3) in order to calculate the context similarity between the tags mentioned in a new question and those in the previous questions answered by the sameuser. To do so, we use the Fasttext model with the pre-trained embeddings on the Common Crawl dataset[4] to obtain embedding vectors for question's tags and tags of the questions answered by the user. Subsequently, we compute this feature for each question and eachuser by computing the cosine similarity of their embedding vectors, as shown in Eq. 2, where V_{q_i} and V_{u_i} represent the embedding vectors of question tags and past questions answered by the user, respectively.

$$tg_7(q_i, u_i) = cosine(V_{q_i}, V_{u_i}), \tag{2}$$

Text Similarity Features. The similarity between past textual content (e.g., answers, and questions) of a user and a new question can be a sign of the relevance of that user to the new question. To capture textual similarities, we define forteen features that compute the similarity between the content posted by a user and the newlyposted question as defined in

[4] https://bit.ly/3oBFTJ0.

Table 3
Content-based features extracted for each answerer given a question.

Feature	Description	Represents
	Question Characteristics Features (QC)	
qc_1	Presence of code in the question (*Boolean*)	Q
qc_2	Length of the question title	Q
qc_3	The number of words in the question title	Q
qc_4	Length of the question body	Q
qc_5	The number of words in the question body	Q
	Tag Features (TG)	
tg_1	The number of tags in the question	Q
tg_2	The number of tags in user’s questions	U
tg_3	The number of tags in past questions answered by user	U
tg_4	The count of common tags between past questions answered by user and the question	QU
tg_5	The count of common tags between user’s questions and the question	QU
tg_6	Tag specificity of questions answered by user	U
tg_7	Embedding similarity between question’s tags and previous questions answered by user	QU
	Text Similarity Features (TX)	
tx_1	Question’s title - all user’s answers	QU
tx_2	Question’s title - all user’s comments	QU
tx_3	Question’s title - questions’ title asked by user	QU
tx_4	Question’s title - questions’ body asked by user	QU
tx_5	Question’s title - questions’ title answered by user	QU
tx_6	Question’s title - questions’ body answered by user	QU
tx_7	Question’s body - all user’s answers	QU
tx_8	Question’s body - all user’s comments	QU
tx_9	Question’s body - questions’ title asked by user	QU
tx_{10}	Question’s body - questions’ body asked by user	QU
tx_{11}	Question’s body - questions’ title answered by user	QU
tx_{12}	Question’s body - questions’ body answered by user	QU
tx_{13}	Average similarity of question’s title and body - all user’s answers	QU
tx_{14}	Overall match - question - all user’s answers	QU
	Readability Features (RD)	
rd_1	Average of user answers’ automated readability index	U
rd_2	Average of user answers’ Coleman-Liau index	U
rd_3	Average of user answers’ Flesch-Kincaid grade	U
rd_4	Average of user answers’ Flesch reading ease	U
rd_5	Average of user answers’ Gunning Fog index	U
rd_6	Average of user answers’ Läsbarhets index	U
rd_7	Average of user answers’ SMOG grade index	U
rd_8	Question’s automated readability index	Q
rd_9	Question’s Coleman-Liau index	Q
rd_{10}	Question’s Flesch-Kincaid grade	Q
rd_{11}	Question’s Flesch reading ease	Q
rd_{12}	Question’s Gunning Fog index	Q
rd_{13}	Question’s Läsbarhets index	Q
rd_{14}	Question’s SMOG grade index	Q
	Topic Features (TP)	
tp_1	Percentage of questions have same dominant topic as the question	Q
tp_2	Percentage of user’s answers have same dominant topic as the question	QU
tp_3	Number of distinct dominant topics in user’s answers topic vectors	U
tp_4	Total score of user’s answers whose dominant topic is the question’s dominant topic	QU
tp_5	Similarity of question topic vector and averaged user’s answers’ topic vector	QU
tp_6	Average of question dominant topic’s value in user’s answers topic vectors	QU
tp_7	Question dominant topic’s value in score weighted average of user’s answers’ topic vectors	QU
tp_8	The entropy of user’s answers’ dominant topic	U
tp_9	The entropy of averaged user answers’ topic vector	U

Table 4. These features are based on the Word Mover’s Distance (WMD) [33], which essentially computes the dissimilarity between two textual contents. WMD computes the minimum distance in embedding space that the words of one text need to travel to reach the words of the other text. To efficiently compute these features, we considered the first 500 words of each textual snippet from the question and user’s most recent content after data cleaning (e.g., removal of stop-words).

Furthermore, we propose a novel feature, denoted as tx_{13}, to compute the average semantic similarity between the posted question and user’s previous answers. For this purpose, we build the embedding vector of the question’s title and body together and each of the user’s previous answers from the Fasttext model trained on the Common Crawl dataset. Then, we calculate the cosine similarity between the embedding vector of the question and each answer. Finally, the average

Table 4
Textual contents that are taken from a user and a question.

Content	
User	Question
User's answers User's comments Question's title asked by user Questions' body asked by user Questions' title answered by user Questions' body answered by user	Question's title Question's body

cosine similarity scores is assigned as the *overall similarity*. Eq. 3 formulates tx_{13} where A is the list of user's previous answers and V_{q_i} and V_a are the embedding vectors of the question and user's answer, respectively.

$$tx_{13}(q_i, u_i) = \frac{1}{|A|}\sum_{a\in A} cosine(V_{q_i}, V_a), \tag{3}$$

Moreover, tx_{14} calculates the **overall match** [34] between the question and the user by counting the number of non-stop terms in the user's answers. This feature intends to measure the relevance of a given question to the answers that a user provided.

Readability Features. The readability of textual content, be it a posted question or a submitted answer, could play an important role in the popularity of the content. Readability metrics measure the difficulty of reading a text by considering its characteristics such as the count of its syllables, words, and sentences. There have been existing work that have utilized readability-based features [23] for predicting the rank of answers before they receive votes. In our paper, we define forteen readability-based features to capture the readability of questions and answers by adopting readability indices including *Automated readability index* [35], *Coleman-Liau index* [36], *Flesch-Kincaid grade* [37], *Flesch reading ease* [38], *Gunning Fog index* [39], *Läsbarhets index* [40], and *Smog grade index* [41] in the context of the question routing problem. To compute the readability score of a question, we considered only the body of the question since the titles usually lack the basic grammatical rules of writing. The readability index of a user is defined over the average of readability score of the user's answers.

Topic Features. Inspired by [10] and to complement tags for capturing a question's content, we build topic model features that represent users, questions and their interactions using topic distribution vectors. Given all questions (consisting of title and body) and answers, we employ the LDA topic modeling technique[42], over the textual corpus after removing symbols and English stop words in order to infer a set of topics. It is worth noting that we assumed thatthe highest value in a topic vector dimension represents the domination of that topic for the content. Eq. 4 shows the formulation of feature tp_1 where *dom(q_i)* is denoted as the dominant topic of question q_i, Q is the set of all questions and n is the number of members of a set. tp_1 captures how prevalent a topic is among all the questions.

$$tp_1(q_i) = \frac{n(\{q_j | q_j \in Q, dom(q_j) = dom(q_i)\})}{n(Q)} \tag{4}$$

Given the topic vectors of a user's answers, Eq. 5 computes tp_2 to capture question-user interaction:

$$tp_2(q_i, u_i) = \frac{n(\{a_j | a_j \in A(u_i), dom(a_j) = dom(q_i)\})}{n(A(u_i))} \tag{5}$$

where $A(u_i)$ is defined as the set of all answers posted by user u_i.

To take a step beyond feature tp_2, tp_4 obtains the expertise level of u_i in the dominant topic of q_i. The score of an answer (i.e., upvotes and downvotes from other users) is taken into consideration as a level of proficiency. Eq. 6 shows the formulation of feature tp_4 where "score" is the linearly normalized difference between downvotes and upvotes of answer a_j. The reason for using the normalized value is to avoid having a zero coefficient for an answer's topic vector.

$$tp_4(q_i, u_i) = \sum score(a_j) | a_j \in A(u_i), dom(a_j) = dom(q_i) \tag{6}$$

A topic vector $Tp(u_i)$ is assigned to each user by averaging the topic vectors of the user's answers. Based on this formulation, the similarity of a user's topic vector and a question's topic vector can be calculated. To compute the similarity between two topic vectors, we employcosine similarity proposed by Riahi et al.[9]. Note that, other similarity metrics can be also deployed for this purpose. Eq. 7 shows how feature tp_5 is computed where $Tp(q_i)$ denotes question q_i's topic vector.

$$tp_5(q_i, u_i) = \frac{Tp(u_i).Tp(q_i)}{|Tp(u_i)| \times |Tp(q_i)|}. \tag{7}$$

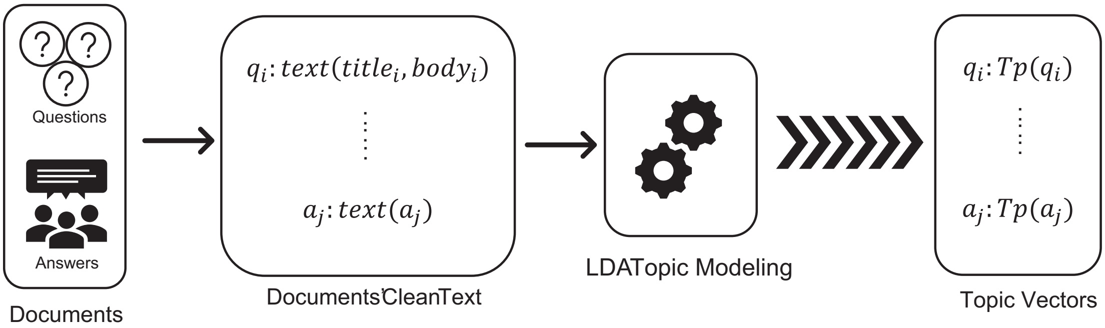


Fig. 4. The development process of topic vectors with LDA.

Following the definition of $Tp(u_i)$, scores of answers are considered as their weights to obtain the score-weighted topic vector of a user, denoted as $ScTp(u_i)$. Furthermore, features tp_6 and tp_7 are proposed to capture the average contribution and score-weighted average contribution of user u_i in the dominant topic of q_i, i.e., $dom(q_i)$, respectively. Eq. 8 and Eq. 9 compute features tp_6 and tp_7, respectively, where $Val(vec)|_{dim}$ indicates the value of a vector vec in dimension (dim).

$$tp_6(q_i, u_i) = Val(Tp(u_i))|_{dom(q_i)} \tag{8}$$

$$tp_7(q_i, u_i) = Val(ScTp(u_i))|_{dom(q_i)} \tag{9}$$

We additionally propose features tp_3, tp_8 and tp_9 to capture the range anddiversity of topics that a user has contributed to. The two features tp_8 and tp_9 are defined based on topic entropy [32]. Feature tp_8 is computed as:

$$tp_8(u_i) = -\frac{1}{|DomTp_{u_i}|}\sum_{j=1}^{|DomTp_{u_i}|} P(topic_j|u_i)ln(P(topic_j|u_i)), \tag{10}$$

where $DomTp_{u_i}$ is a vector of dominant topics of answers in $A(u_i)$ (set of answers of user u_i) and $P(topic_j|u_i)$ is defined as the probability of $topic_j$ to be in $DomTp_{u_i}$.

Similarly, we employ feature tp_9 to capture the entropy of topics in user's topic vector, which is computed as follows:

$$tp_9(u_i) = -\frac{1}{|Tp_{u_i}|}\sum_{j=1}^{|Tp_{u_i}|} P(topic_j|u_i)ln(P(topic_j|u_i)). \tag{11}$$

4.2.2. Social-based features

In the second category of features, we define social-based features that capture aspects of a CQA platforms that relate to the users, their temporal activities and community network structure. We summarized the proposed social-based features in Table 5. Social-based features are categorized into three subcategories, namely user characteristics, user temporality, and network characteristics.

User Characteristics. Similar to question characteristics, we hypothesize that there are certain user-related attributes that can serve as prior probability of a user beingpotentially interested in answering a question. We define features to quantify such prior probability for each user. Features such as user's reputation score or the number of past best answers on the CQA platform are indicators of how engaged the user is in the community and thus, can help measure the likelihood for that user contributing to anewly posted question. We define nine features based on user characteristics, which will serve as indicators of the probability of the user answering a posted question regardless of the contents of that question. Note that, user score is calculated based on the upvotes and downvotes thata user has received on her answers while reputation is calculated based on different factors such as votes on answers, questions or comments, editing an answer, and the number of best answers.

Inspired by [34], we define **average span**, uc_{10}, to measure the conciseness of a user's previous answers by calculating the largest distance between two non-stop question's terms in the user's answers. In order to compute this feature, we calculate the span of every two terms in the question in each of the user's previous answers and take the average over all of the answers. We also compute **informativeness** inspired by [43], uc_{11}, to measure the amount of information offered by a specific user's answers. This is accomplished by calculating the term frequency of non-stop nouns, verbs, and adjectives in the answers' text provided by the user, which do not appear in the title or body of thequestion.

User Temporality. We define temporal characteristics for a user based on her availability to answer a newly posted question by considering the user's past availability. It has already been argued in the literature that a user is more likely to answer a question if she (1) is active on the CQA platform when the question is posted [8,14], and (2) has shown active history of participation on the CQA platform in the past [44]. Existing works in the literature [14,8,9] have reported that time is a popular measure for question routing. Here, we defined five features to represent user temporality. These features are

Table 5
An overview of the proposed social-based features.

Feature	Description	Represents
	User Characteristics (UC)	
uc_1	The number of questions posted by the user	U
uc_2	The number of answers submitted by the user	U
uc_3	The number of user comments	U
uc_4	User's reputation	U
uc_5	If the user has a website (*Boolean*)	U
uc_6	The number of user profile views	U
uc_7	The user's score	U
uc_8	The number of best answers posted by the user	U
uc_9	The difference between the number of user's answers and user's questions	U
uc_{10}	Average span of user's previous answers	U
uc_{11}	Informativeness of user's answers	U
	User Temporality (UT)	
ut_1	Availability of the user for the question during the day by hour	QU
ut_2	Difference between user's creation time and user's first best answer's time	U
ut_3	Median of interval time of user's best answers	U
ut_4	Difference between user's creation time and user's first answer's time	U
ut_5	Median of interval time of user's answers	U
ut_6	Availability of the user for the question during the week by day	QU
ut_7	Availability of the user for the question during the year by month	QU
	Network Characteristics (NC)	
nc_1	User has answered at least one question from asker (*Boolean*)	QU
nc_2	Number of previous asker's questions answered by the user	QU
nc_3	Cosine similarity of Node2Vec(Question) and Node2Vec(User)	QU
nc_4	User Degree centrality	U
nc_5	User Closeness centrality	U
nc_6	User Eigen centrality	U
nc_7	User PageRank measure	U

especially important as they can rule out the possibility of recommending a user with the right skillset and interests while she is not available on the platform or often has a high delay in providing answers.

To capture the availability of a user, similar to [14], we estimate time distribution for each user based on the timestamp of their pastactivities. We collect the timestamps of when questions, comments, and answers of a user have been posted in orderto generate a contribution time distribution for each user. Fig. 5 illustrates contribution time distribution of a user, as an instance. The figure shows that this specific user is mostly active on the CQA platform during the evenings and hence has a higher likelihood of answering questions during this time. If a question is recommended to this user between 7am-3 pm, it is unlikely that they would respond immediately. On this basis, given a new question, ut_1 indicates the probability of the user being online at the time the question is posted.

In addition to the availability of a user during a certain hour of a day, we also explore the chance of user availability with respect to the day of the week and month of the year. As a result, we define features ut_6 and ut_7, respectively. Based on these two features, our proposed model leverages the probability of a user being available during specific days of the week or months of the year and thus prioritizes users with higher likelihood of being available to answer a question. Similar to ut_1, ut_6 and ut_7 are also indications of users' availability to answer a newly posted question.

Network Characteristics. The past interactions between users involved in the question answering process (e.g., askers, answerers) in a CQA platform can serve as an indicator of future interactions. For instance, one could assume that if a user has consistently answered several questions posted by another user, it might be likely that she will alsoanswer a future question posted by that same user in the future. In order to capture such relationships, we develop a heterogeneous network, as shown in Fig. 6, in which the nodes are questions, askers, and answerers, and the edges are question-asker, question–answerer, and answerer-answerer relationships. The edge weight between a question and its asker is computed based on the difference between the number of upvotes and downvotes on the question. Furthermore, the weight of the edge between a question and the users who have answered this question is also similar, which is equivalent to the difference between the upvotes and downvotes on the answer. The weight of edges between users is computed as the fraction of the number of common questions answered by two users divided by the total number of answers provided by both users in the community, Eq. 12.

$$w(u_i, u_j) = \frac{|A(u_i) \cap A(u_j)|}{|A(u_i) \cup A(u_j)|} \tag{12}$$

We leverage graph-based measures such as degree centrality, network closeness centrality, and Eigen centrality, to name a few, directly from the constructed graph to compute network-based features for users or question-user pairs. The set of seven network characteristics is shown in Table 5. We additionally point to feature nc_3 as it performs soft matching between

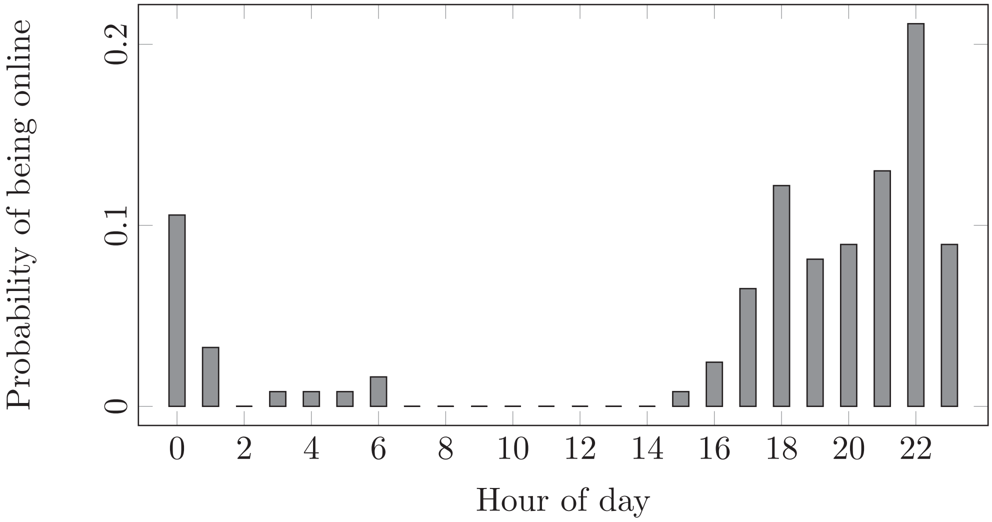


Fig. 5. Contribution time distribution of a sample user.

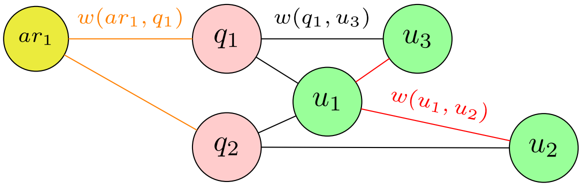


Fig. 6. Network of questions (q_i), answerers (u_i), and askers (ar_i). The weight of the edge between $node_1$ and $node_2$ is shown as $w(node_1, node_2)$. Each edge in the graph has a non-zero weight. For simplicity, all weights are not shown.

questions and experts. This is done through the extraction of a **neural embedding** representation for graph nodes, using the node2vec method [45], based on which cosine similarity between question and user nodes (vectors) is computed. Eq. 13 formulates the nc_3 feature as follows:

$$nc_3(q_i, u_i) = \frac{\overrightarrow{nod2vec}(u_i).\overrightarrow{nod2vec}(q_i)}{|\overrightarrow{nod2vec}(u_i)| \times |\overrightarrow{nod2vec}(q_i)|} \tag{13}$$

where $\overrightarrow{nod2vec}(node_i)$ is the vector of $node_i$ from the node2vec representation of the graph.

5. Experimental evaluation

In our experiments, we investigate 1) the impact of each feature category on the performance of our proposed question routing approach; 2) the performance of our proposed approach in comparison with the state-of-the-art baseline question routing techniques; and 3) the importance of individual features in the performance of the overall question routing approach. To this end, we have carried out extensive experiments on five real online datasets, which are among the largest CQA datasets available. We present our observations and findings in the following subsections.

5.1. Datasets

We have used the data collected from five popular question answering websites published by Stack Exchange.[5] These datasets are publicly accessible online at Stack Exchange archives.[6] The five datasets include *Super User*,[7] which is an online

[5] https://stackexchange.com/.
[6] https://archive.org/details/stackexchange.
[7] https://superuser.com/.

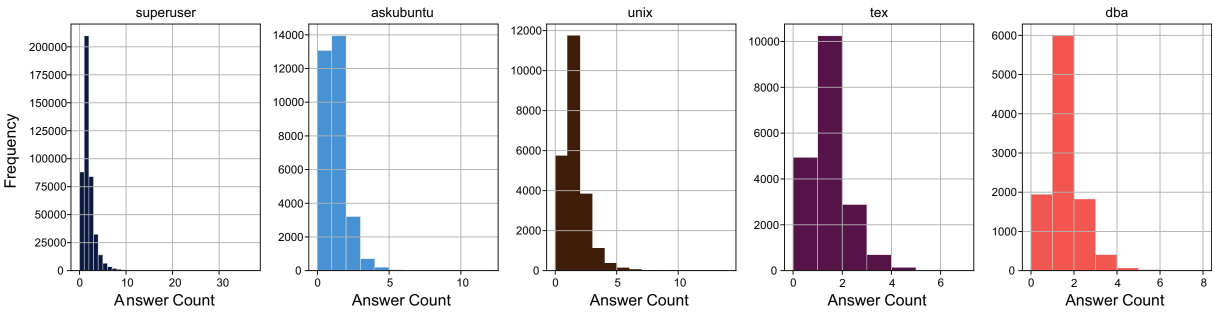


Fig. 7. Distribution of answer counts provided per question in different datasets.

CQA website for computer enthusiasts and power users hosted by Stack Exchange; *Tex*[8] that is a question answering platform for users of Tex, Latex, ConteXt, and related typesetting systems; *Askubuntu*,[9] which is a website for Ubuntu users and developers; *Unix* & *Linux*[10] that addresses questions related to Linux, FreeBSD and other Unix-like operating systems; finally, *Database Administrators*,[11] referred to as *dba*, consists of questions posted by database professionals who wish to improve their database skills and learn from others in the community. Fig. 7 shows the number of questions in these datasets grouped by the number of answers they received. The gap in having questions answered is a clear indication for the need to address this problem. In all datasets, there are least 20% of the questions that remain unanswered. In *askubuntu*, we have the largest gap with over 42% of the questions being unanswered.

In our experiments, we pre-processed each dataset by removing any stop-words, non-English content, and special characters, among other pre-processing steps. In order to prepare the training data, we extracted question threads posted in 2019, which had at least two answers. For the test data, we collected question threads which were posted in the first month of 2020. Furthermore, we onlykept those users in the test set who had answered at least one question in the training set. This is due to the fact that these users can be the potential answerers in the future and also the computation of some of ourfeatures require the history of user activities. For each question thread (either in the train or test data), we added random users for the sake of negative sampling. The number of such negative samples was the same as the number of true answers of questions with ten or fewer answers and was ten for the others. Note that, to train an LTR model, we have to provide negative samples representing users who are not the answerers of a given question in the training set. This provides the knowledge for the method to avoid bias and train the model on irrelevant users who need to be avoided. In the test and the train data, for each question, the answerer with the accepted answer receives the highest ranking score and other answerers were assigned linearly decreasing scores (the lowest score is one) based on the answer votes. For a question with no accepted answer, the answer with the highest voting score was treated as the accepted answer. The negative samples received a ranking score of zero.

We further note that models are trained based on the question-user pairs extracted from the dataset. Given each question in the dataset, we select top-k answerers as related documents to the input question and we randomly selected k users from the dataset as unrelated documents to the question. The pairs have been passed to different algorithms to train the model. Table 6 shows a brief piece of information about each dataset.

5.2. Experimental setup

In order to train the learn to rank models, we use nine different learn to rank methods covering pair-wise, point-wise and list-wise learn to rank models and compare the performance of our proposed features over these different learn to rank methods. The implementation of these methods are based on the default parameters provided by the RankLib[12] software package, a commonly used open source learning to rank library. Furthermore and without loss of generality, in order to measure feature importance, We utilized Random Forests, which allow us to compute Gini score for each of the features.

We used **Normalized Discounted Cumulative Gain** at top k (NDCG@k) and **Mean Average Precision** at top k (MAP@k) as the ranking measures in this work. Also, note that we built the required LDA topic models [42] using the Mallet library[13] over all questions and answers in the training dataset. The best topic size was chosen to be 100 for the superuser dataset and 50 for other datasets based on the *coherence value*, a common metric for evaluating topic model quality, which has also been used by [46]. We select the best topic size by exploring different topic set sizes ranging from 50–350 as shown in Fig. 8. To calculate the network characteristics features, we used the networkx library.[14] For computing text similarity features, we used pre-trained FastText embeddings trained on Wikipedia [47].

[8] https://tex.stackexchange.com/.
[9] https://askubuntu.com/.
[10] https://unix.stackexchange.com/.
[11] https://dba.stackexchange.com/.
[12] https://sourceforge.net/p/lemur/wiki/RankLib/.
[13] https://mimno.github.io/Mallet/.
[14] https://networkx.org/.

Table 6
Statistical information of the datasets.

Dataset	Questions (Q)	Answers	Tags	Users	# training Q
superuser	5,183	11,795	1,918	4,916	4,885
askubuntu	4,245	9,939	1,334	4,923	4,040
unix	5,564	13,757	1,349	3,358	5,157
tex	3,762	8,564	944	1,123	3,510
dba	2,338	5,261	628	1,237	2,176

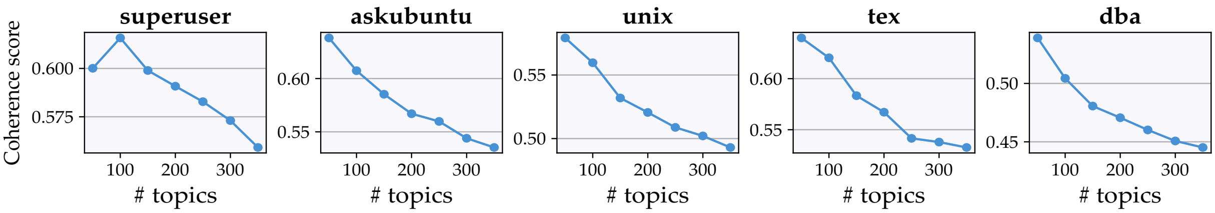


Fig. 8. Coherence value of the different number of topics over each dataset.

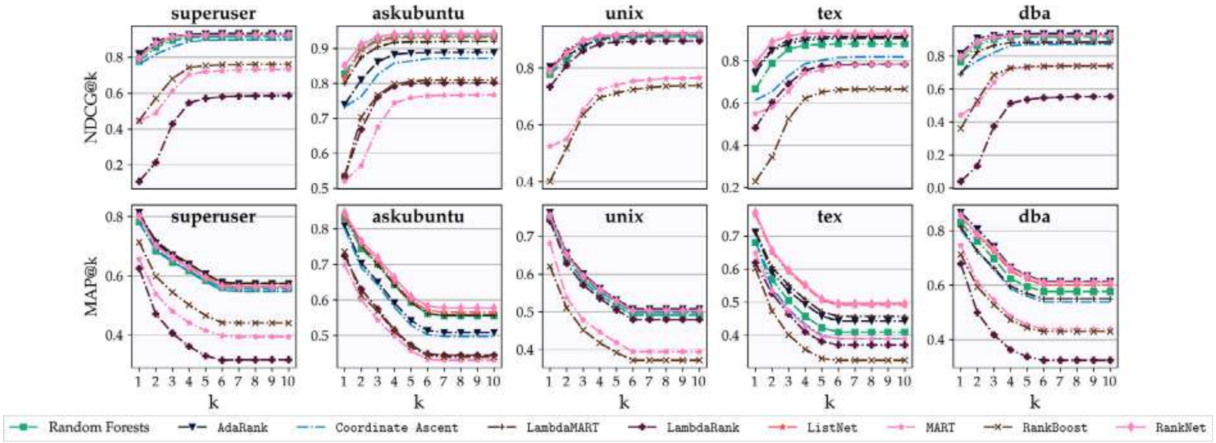


Fig. 9. Comparison of the different LTR models in terms of ranking metrics NDCG@*k* and MAP@*k*.

5.3. Baselines

We compared the performance of the proposed approach to the state-of-the-art (SoTA) methods, namely `DUET` [20], `ConvKNRM` [18], `DSSM` [19], and `ANMM` [21], and the most recent neural model for question routing with SoTA performance, i.e., `Seq` [4]. The learning to rank methods utilize the textual information of past questions (tags, title, and body) and answers to pair users with new questions. First, they learn the latent representation of the content for each question and its answers. Then, the model is used to predict the relevance of each potential answerer to thenew question based on the content of the user's past answers. Additionally, `Seq` [4] uses the structural information of the network constructed using questions, tags, askers, and answerers to rank users for a new question. First, it learns latentrepresentations and then uses the these representations to match users to new questions. We present the results of comparing our work with the baselines in Section 5.8.

5.4. Comparison of different LTR methods

In this section, we investigate the performance of different LTR methods on the performance of our proposed work. Fig. 9 shows that the majority of the models are very competitive except MART, Rank Boost and LamdaRank. Among the nine models, MART is a point-wise, RankNet and RankBoost are pair-wise, and Random Forests, AdaRank, Coordinate Ascent, LambdaMart, LambdaRank and ListNet are list-wise LTR models. From the high-performing models, we chose Random Forests as the LTR model in the remainder of our experiments since this model allowed us to perform a Gini importance analysis. Gini importance is a common tool to analyze and interpret feature importances, which is one of the main objectives of our work in this paper. We note that the choice of the LTR method does not impact the overall performance of our features, i.e., other competitive LTR methods would show similar performance over our features.

5.5. Impact of content-based and social-based features

We performed experiments to understand the influence of content-based and social-based features and their combination on the question routing task. We trained the LTR models for each of the feature sets, separately. The results, in terms of the evaluation metrics, i.e., NDCG (first row) and MAP (second row), across a range of sizes for *k* over all the datasets, are reported in Fig. 10. The figure shows that regardless of the search depth *k*, content-based features are more effective compared to social-based features. We especially observed that social-based features are not only weaker than content-based features but are also detrimental to the overall performance of the learning to rank model. Although the difference in the performance level of content-based and social-based features is different over the different datasets, their order of strength (i.e. Content-based, All, Social based) is consistent in all of the five datasets.

We are further interested to understand the impact of each subcategory of features on the overall performance of the question routing task. This is especially important because there have been works in the literature that showed various social network-based features are effective for downstream various tasks such as answer ranking and expert finding. This is in contrast to our observations where a negative impact on performance was seen. We will explore this in the next subsection.

5.6. Analysis of individual feature sub-categories

To analyze the impact of each feature sub-category on the performance of the model, we used the features of each subcategory separately to train LTR models. The results over all datasets are reported in Fig. 11 based on the evaluation metrics, i.e., NDCG and MAP respectively. This experiment provides interesting insights, which we outline as follows:

(a) In all five datasets, there werefour feature sub-categories that showed to be the strongest, namely: user characteristics, tag, topic modeling and user temporality. Having user characteristics on this list was not a surprise as this subcategory specifically targets the differences between users. Furthermore, the results indicate that congruence between the new question's tags and the tags associated with questions that the user had a past engagement with is critical for question routing. It is also noted that another well-performing feature sub-category in content-based features is the topic feature sub-category. One could, in essence, view topic features to be very similar to tag features with the difference that tags are explicitly assigned by the users while topics are learned implicitly from the content. In our experiments, we found that tags are more effective than topics, although both feature sub-categories remain strong and competitive. In addition, readability features appeared in the 3^{rd} place on thedba dataset but its performance was not consistent on all datasets.

(b) While we find thatsocial-based features are detrimental to the overall performance of the question routing task, this seems to have been the result of the poor performance of the network characteristics features. Fig. 11 displays the individually weak performance of network characteristics in all datasets. Within the social-based features, user characteristics and user temporality feature subcategories are the best performing feature sub-categories. The good performance of user characteristics means that the prior probabilities indicating the likelihood of a user engaging with a new question

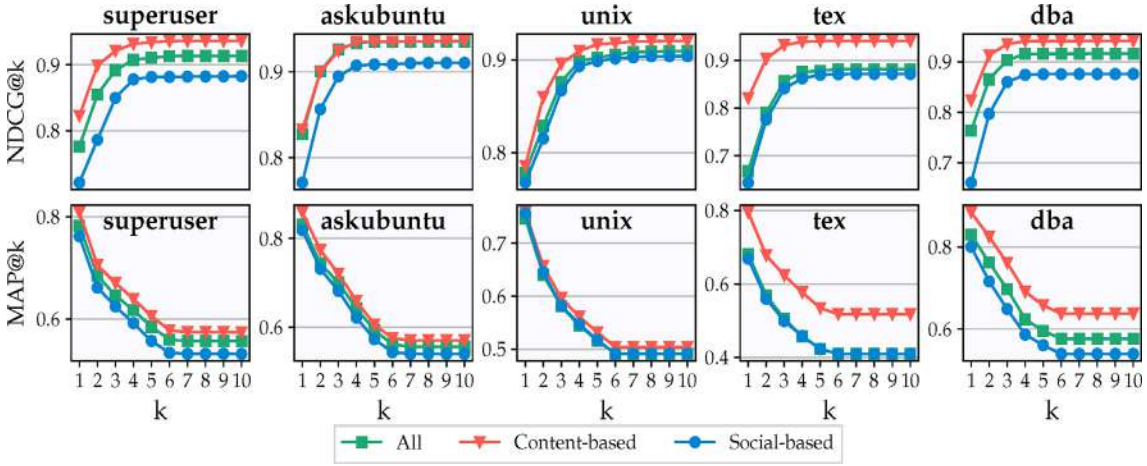


Fig. 10. Analysis of the model's performance over all datasets when using content-based, social-based, or all features based on NDCG@*k* and MAP@*k*.

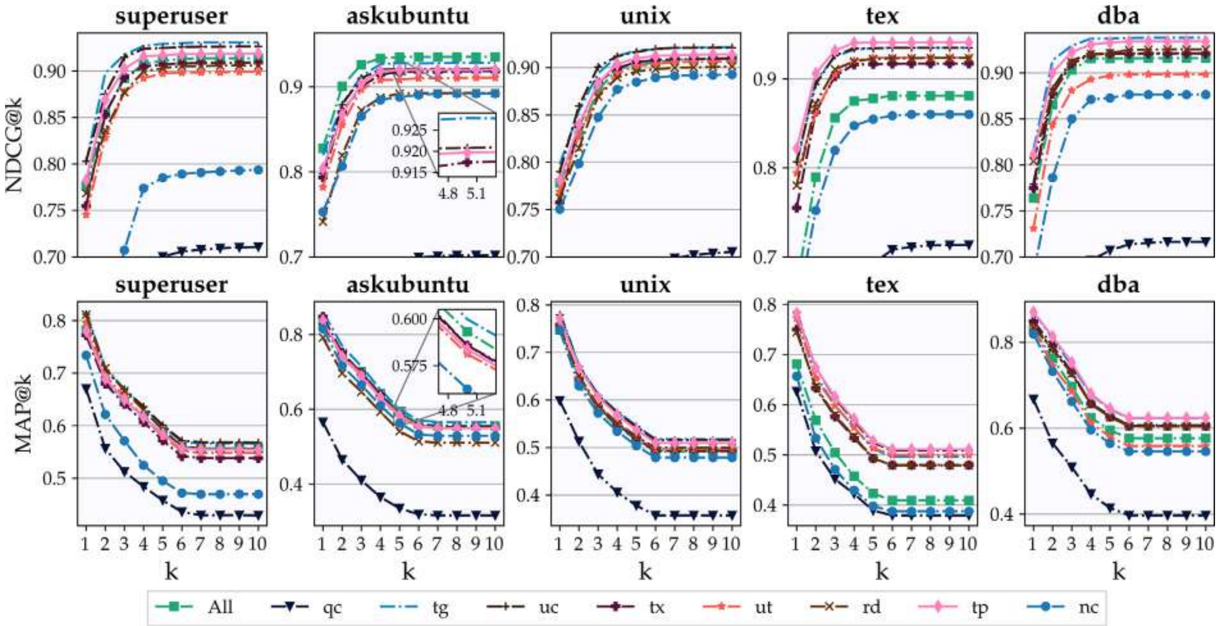


Fig. 11. Analysis of model's performance over all datasets when using each individual feature class based on NDCG@*k* and MAP@*k*.

built solely based on the user's past participation record is a strong indicator. This shows thatactive users in the CQA platform are those that are likely to engage with future questions. The performance of the user temporality feature subcategory also indicates that users who are available on the network when the question is first posted are more likely to answer it. This reinforces earlier findings that a question on CQA is either answered in the first hour or it becomes less likely to receive an answer in the future [48].

(c) The weakest performing feature sub-category is related to question characteristic features (QC). This was anticipated as question characteristic features areonly able tocapture aspects of thequestion itself and do not draw a relationship to the user space to be used for ranking the users. The finding is similar to observations made in learning to rank approaches used for other tasks where query features have shown not to be strong features [49].

5.7. Ablation analysis of feature sub-categories

We have conducted an ablation study to reveal the impact of each individual sub-categories of features on the performance of the model. As such, we trained models using all the features defined in this study and subsequently exclude individual sub-categories from it. The results of the ablation study are reported in Fig. 12. We made several observations as follows:

(a) One of our main observations based on the result of the ablation study, which gradually removes feature subcategories, corroborates the findings in Fig. 11. This finding indicates thatthe four most important feature subcategories identified earlier are tag, user characteristics, user temporality and topics features. Tag related features proved to be the most vital feature class in the model for all the datasets in order to have a strong performance. Other feature sub-categories did not appear consistently through all datasets.

(b) We found network characteristics (NC) features to remain to be the poorest feature sub-categories. As shown in the figure, their removal from the model actually leads to better performance in all datasets compared to when all features were present.

Given the poor performance of network characteristic features, it is important to further understand whether these features are generally not appropriate for question routing, or there are specific features within this feature sub-category that are more effective compared to the rest. As such, we performed an additionalablation study specifically on each of the network characteristics features. The results are shown in Fig. 13. We observed the best performance of the proposed LTR approach on all five datasets when network characteristics features were fully removed from the features. Our findings confirm those by [50] that people are more likely to consider the content of a question to answer it rather than their relationship with the owner of the question.

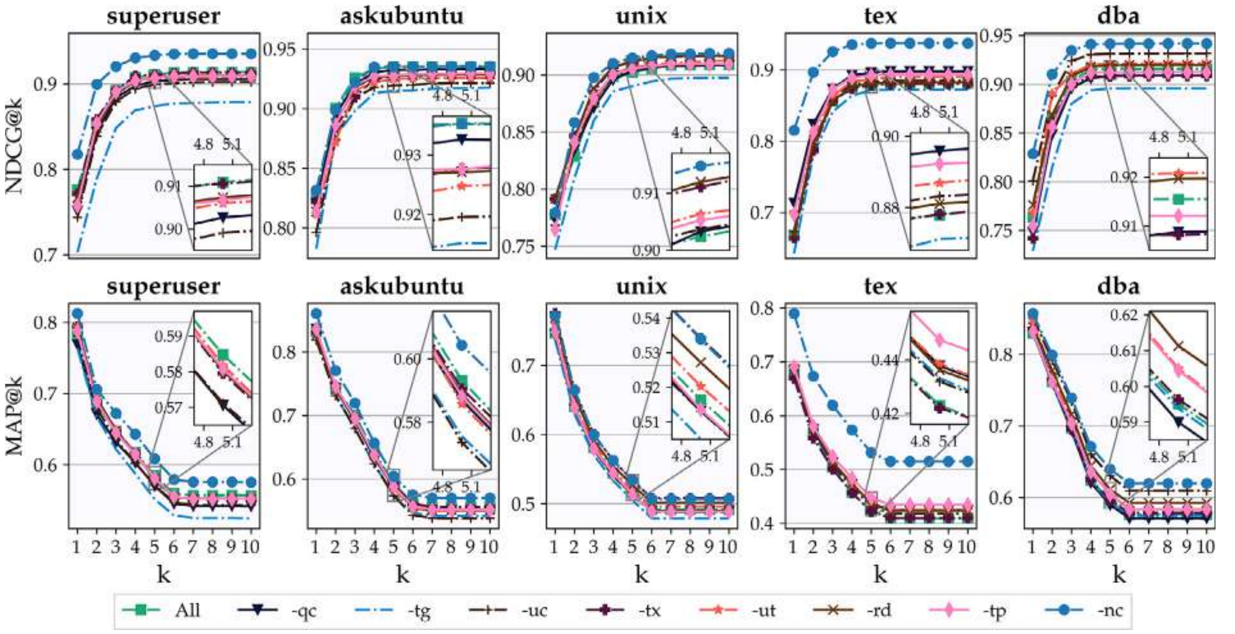


Fig. 12. Analysis of model's performance over all datasets when each individual feature class is ablated from the set of all features based on NDCG@*k* and MAP@*k*.

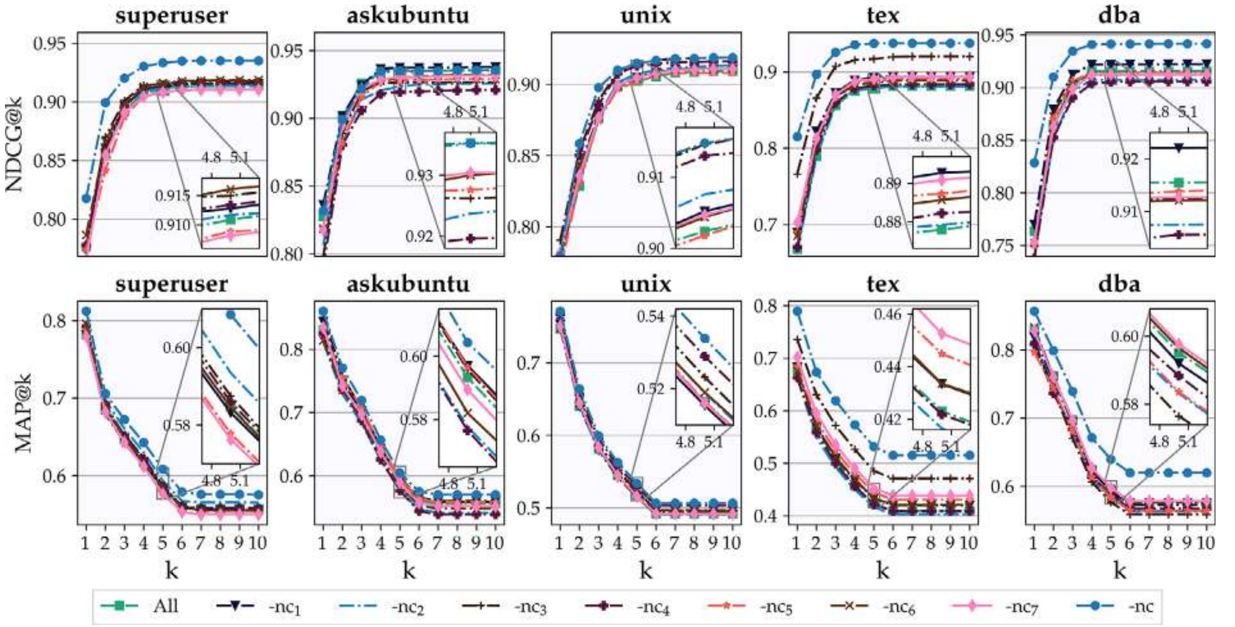


Fig. 13. Analysis of model's performance over all datasets when each network feature is ablated from the "all features" set based on NDCG@*k* and MAP@*k*.

5.8. Comparison with baselines

As outlined in the introduction of this paper, while this work attempts to provide a transparent interpretation of feature suitability for question routing, at the same time, it is important to show that these features are able to perform competitively to or better than the state-of-the-art.

To do so, we compared the performance of our model with the state-of-the-art baselines introduced in Section 5.3. The baselines were applied on the fivedatasets whose results are reported in Fig. 14. As shown in this figure, using all of our pro-

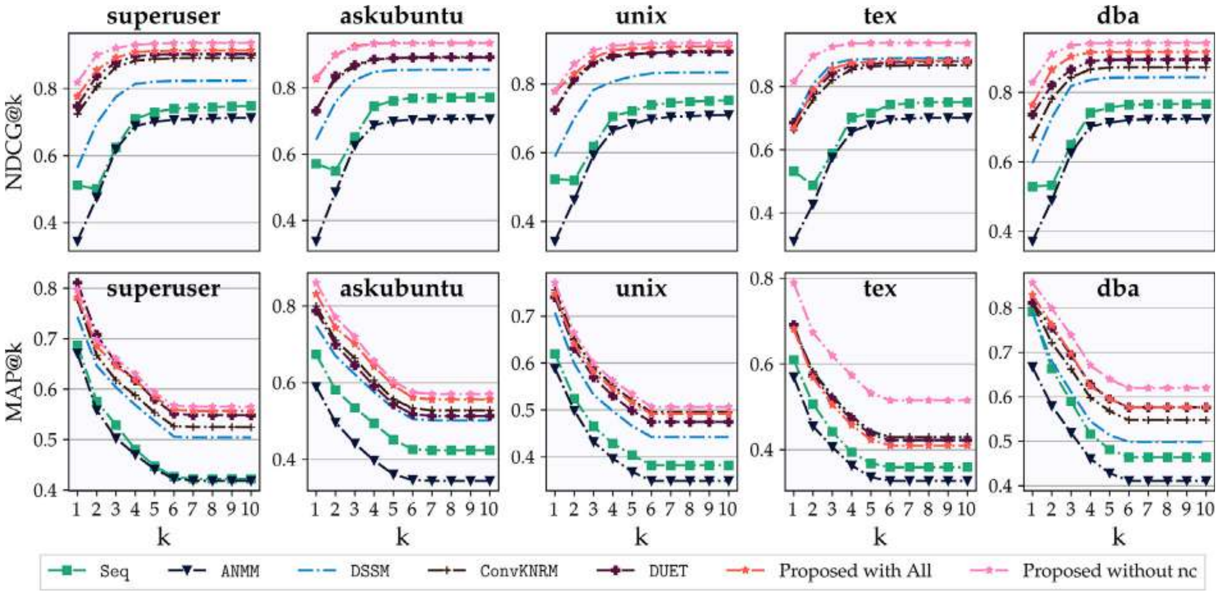


Fig. 14. Comparison of the proposed approach with state of the art baselines over all datasets in terms of ranking metrics NDCG@*k* and MAP@*k*.

posed features, our model outperforms almost all of the baselines on all datasets. We note that when excluding network characteristics features, we are able to see more accurate results compared to when they are included.

As shown in the figure, the learning to rank model trained based on the proposed features is able to outperform the baselines in terms of both ranking metrics, NDCG (first row) and MAP (second row). The proposed model without the network features outperformed the best baselines by a minimum of 2.47% in NDCG and 1.10% in MAP compared to DUET and ConvKNRM, respectively. On average, on all datasets, our proposed approach outperformed DUET, the best baseline in the experiments, by 4.10% in NDCG and 5.04% in MAP. From among the baselines, the most recent baseline that is specifically designed for question routing, i.e., seq, did not show the best performance compared to the other baselines. Our proposed approach showed a minimum of 16.41% and 12.47% improvement, over seq on NDCG and MAP@$k = 10$, respectively. We conclude that the set of features proposed in our paper is able to not only provide transparency into the utility of features for thequestion routingtask, but also to provide better performance compared to the state of the art baselines including the most recent work in this area, namely seq [4].

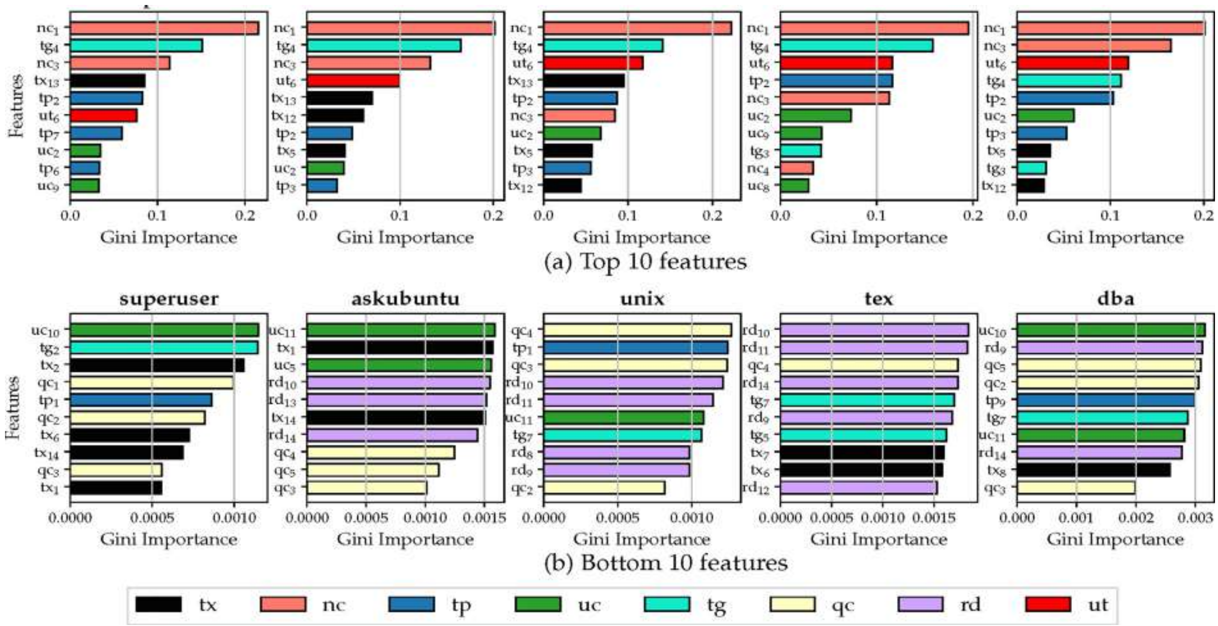
5.9. Feature analysis using Gini importance

To analyze the importance of each individual feature, we employed the Gini importance criterion. Gini importance can indicate the level of impact each feature has on building the question routing model. In a random forests model, each feature is used to split the dataset in order to decrease the Gini impurity. Gini is a measure to evaluate the impurity of a node. Generally, when a feature is used to split a node, it makes two nodes with less Gini impurities ($Gini_{c_1}$ and $Gini_{c_2}$). Gini impurity of the children nodes is the weighted average of the Gini impurity in each child node. The difference in impurity ($\Delta Gini_{f_i}$) of the parent node ($Gini_p$) and the weighted average of children nodes is considered as the impact of that feature. In other words, the change in impurity is considered to be the importance of that certain feature in that node. A feature could be used in different nodes of a tree and different trees of a random forests model. The weighted average of all the importance values of a feature over all nodes and trees is the importance of that feature. The Gini importance of feature f_i is computed as:

$$GiniImportance(f_i) = \Delta Gini(f_i) = Gini_p - \frac{n_1 \times Gini_{c_1} + n_2 \times Gini_{c_2}}{n_1 + n_2} \tag{14}$$

We computed the Gini importances over the 74 features proposed in this study. Fig. 15 displays the Gini importance of features grouped in top 10 and bottom 10 based on their level of importance. Through this analysis, we can obtain better insight into the impact of each individual feature.

The nc_1 feature with Gini importance of at least 0.195 in all datasets (see Fig. 15 first row) shows the highest impact on the ranking of users for a given question. Note that Gini importance does not necessarily mean nc_1 will always increase the performance of the model. For instance, Fig. 13 shows that removing nc_1 from the model in some datasets will decrease the performance but will boost it in other datasets in terms of MAP values and will boost the performance in all datasets in terms of NDCG. That being said, if a user has answered a question from the same asker in the past (i.e., $nc_1 = 1$, Table 5), the nc_1 feature would play an important role in determining the ranking of the user to answer a new question of the same asker.



(a) Top 10 features

(b) Bottom 10 features

Fig. 15. Top 10 and bottom 10 impactful features on the model for all datasets based on their Gini importance.

Other features such as tg_4, nc_3, tp_2 and ut_6 were highly consistent in their Gini importance ranking. The Presence of user feature uc_2 in this figure implies that the number of answers in a user's profile is influential when performing question routing.

Fig. 15 (see thesecond row in the figure) displays the 10 least influential features in the ranking model for each dataset. We find that *readability* and *question characteristics* features are occupying the lowest positions based on Gini importance. It is not a surprise that the features representing questions (Q-representative) are able to contribute less in distinguishing the relevance of users for question routing. Question characteristics and readability features (the ones that appear in Fig. 15) are all representing questions only. That does not necessarily mean that the existence of these features will cause a decrease in the performance of the model. As shown in the ablation of feature sub-categories in Section 5.7, removing the above feature classes may cause a decrease in performance.

5.10. Summary of findings

The work in this paper was based on three main objectives (O1-3) as presented in Section 1 which focused on the following: **(O1)** the introduction of features for question routing for which 74 features were introduced and systematically classified as shown in Fig. 3, **(O2)** shedding light into the effectiveness of the proposed features for the question routing task, which allows for a deeper understanding of the features and for the design of mechanisms that encourage question answering. This has been performed based on the analysis of features at category, sub-category levels, as well as, through ablation studies in Section 5.5, and **(O3)** demonstrating that the proposed features are able to show competitive or better performance compared to the state of the art end-to-end neural ranking methods, whose results are provided in Section 5.8.

Summarily, the findings of our paper are as follows:

(a) Users are most inclined to engage in answering new questions if these questions have similar tags or topics to those that they have answered in the past. In other words, the topical experience of users determines the likelihood of their participation in the future;

(b) Even without having information about the specific new question, it is possible to predict which users will engage with a new question by solely relying on observations of the user's past contributions in the CQA platform. The more active theusers are in the past, the more they are likely to contribute in future questions. This combined with the chance of the user being available on the platform when a question is posted form strong indicators for a user's engagement with a question;

(c) Network characteristics features introduced in this paper did not showa positive impact on the question routing task and collectively resulted in a decline in overall performance. In both the individual performance and ablation analysis, network characteristic features proved to have a negative impact on the performance of the model. These results could suggest that for the question routing problem, it is beneficial to disregard the network characteristics between users based on their past activities. Alternatively, as the analysis showed, the emerging solutions can focus on the comparison of the new questions' contents and users' past activity. We further note that focusing solely on features of a posted question would not result in effective performance.

6. Implications

The ever increasing usage of community question answering platforms indicates their popularity and efficacy in practice. The existence of such platforms introduces new challenges that need to be addressed to keep these online communities alive and efficient for knowledge sharing/seeking. Solving the question routing problem seems to be a promising solution for this purpose. Results of the proposed approach show the capability in improving the question routing methods. Also, insight taken from the interpretation of features in this study can help scholars in their future research to emphasize on more impactful and positively effective features.

One of the practical implications of our work in this paper is that CQA platforms can leverage the four most effective feature sub-categories proposed in this study to effectively route new questions to community experts resulting in receiving more reliable answers. Our work shows that considering the topical expertise of users, whether determined by question tags or topic modeling techniques, is beneficial as supported in previous studies such as [14,9]. Furthermore, we found that the activity level of users and also their activity within time can be effective in question routing. On the other hand, our results suggest that network characteristics features are not advantageous for building an effective question routing system. By categorizing features and interpreting their impact on question routing task, we shed light on this area of study to reveal what aspects of the CQA platforms will potentially help scholars and platform administrators toward building effective question routing methods.

Another practical implication of our work is the possibility to use the proposed methodology to build systems for other application domains. For example, our findings can be used to effectively route submitted journal and conference manuscripts to the right reviewers.

7. Conclusion

In this paper, we have adopted a learning to rank approach for the purpose of question routing. We have introduced 74 features and systematically classified them into two categories, namely content and social based, and eight subcategories, namely question characteristics, tag, text similarity, readability, topic modeling, user characteristics, user temporality and network characteristics. The results of our experiments on five real online platforms' datasets show that our proposed approach, enriched with the extensive set of features, is able to outperform the state-of-the-art baselines, including neural methods, on the question routing task. Besides the higher performance, our proposed approach is *interpretable* and *transparent* in determining the effective features.

We show that content-based features can perform effectively when routing questions but social-based features are not as effective in comparison. With further analysis, we showed that network characteristics features are in fact detrimental to the effectiveness of the question routing task in the context of our datasets. User features, temporality features, topic, and tag features, on the other hand, showed to be effective for improving the performance of the question routing task.

We conducted a deeper analysis over network features to identify which features of this class are detrimental to performance. We found that the removal of all network characteristic features results in improved model performance. In addition, based on Gini importance, we showed that the most influential features in the model are user features, topic and tag features. In contrast, question features and readability features do not contribute much to the performance.

Overall, we have shown that it is possible to build transparent and interpretable features within an LTR framework that can additionally outperform the state of the art neural baselines. The insight provided by our work in this paper can aid mechanism design for user engagement in CQA platforms.

As future work, we will consider other perspectives on the analysis of the features. For example, we will utilize different methods of text similarity to analyze their impact on performance and investigate which similarity method outperforms the others in the CQA context. This comparison between models within a certain sub-category can be extended to other features, such features based on topic models and network characteristics, as well. We will also compare the performance of our model with other graph embedding representations, which might enable us to capture more semantically rich features.

CRediT authorship contribution statement

Soroosh Sorkhani: Writing – original draft, Methodology, Software, Investigation, Visualization. **Roohollah Etemadi:** Writing – review & editing, Methodology, Software, Investigation, Visualization. **Amin Bigdeli:** Methodology, Software, Investigation, Visualization. **Morteza Zihayat:** Supervision, Conceptualization, Methodology, Writing – review & editing, Funding-acquisition. **Ebrahim Bagheri:** Supervision, Conceptualization, Methodology, Writing – review & editing, Funding acquisition.

Data availability

Data will be made available on request.

Declaration of Competing Interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

[1] N. Li, B. Guo, Y. Liu, L. Yao, J. Liu, Z. Yu, Askme: joint individual-level and community-level behavior interaction for question recommendation, World Wide Web 25 (1) (2022) 49–72.

[2] M. Neshati, Z. Fallahnejad, H. Beigy, On dynamicity of expert finding in community question answering, Inf. Process. Manage. 53 (5) (2017) 1026–1042.

[3] Z. Li, J.-Y. Jiang, Y. Sun, W. Wang, Personalized question routing via heterogeneous network embedding, in: Proceedings of the AAAI Conference on Artificial Intelligence, vol. 33, 2019, pp. 192–199..

[4] J. Sun, J. Zhao, H. Sun, S. Parthasarathy, Endcold: An end-to-end framework for cold question routing in community question answering services, in: Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence (IJCAI-20), 2020, pp. 3244–3250.

[5] X. Wang, C. Huang, L. Yao, B. Benatallah, M. Dong, A survey on expert recommendation in community question answering, J. Comput. Sci. Technol. 33 (4) (2018) 625–653.

[6] M. Du, N. Liu, X. Hu, Techniques for interpretable machine learning, Commun. ACM 63 (1) (2019) 68–77.

[7] X. Liu, W.B. Croft, M. Koll, Finding experts in community-based question-answering services, in: Proceedings of the 14th ACM international conference on Information and knowledge management, 2005, pp. 315–316.

[8] B. Li, 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, 2010, pp. 1585–1588.

[9] F. Riahi, Z. Zolaktaf, M. Shafiei, E. Milios, Finding expert users in community question answering, in: Proceedings of the 21st International Conference on World Wide Web, 2012, pp. 791–798.

[10] I. Szpektor, Y. Maarek, D. Pelleg, When relevance is not enough: promoting diversity and freshness in personalized question recommendation, in: Proceedings of the 22nd international conference on World Wide Web, 2013, pp. 1249–1260.

[11] Z. Ji, B. Wang, Learning to rank for question routing in community question answering, in: Proceedings of the 22nd ACM international conference on Information & Knowledge Management, 2013, pp. 2363–2368.

[12] L. Luo, F. Wang, M.X. Zhou, Y. Pan, H. Chen, Who have got answers? growing the pool of answerers in a smart enterprise social qa system, in: Proceedings of the 19th international conference on Intelligent User Interfaces, 2014, pp. 7–16.

[13] A. Pal, F. Wang, M.X. Zhou, J. Nichols, B.A. Smith, Question routing to user communities, in: Proceedings of the 22nd ACM international conference on Information & Knowledge Management, 2013, pp. 2357–2362.

[14] S. Chang, A. Pal, Routing questions for collaborative answering in community question answering, in: 2013 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2013), IEEE, 2013, pp. 494–501.

[15] J. Guo, S. Xu, S. Bao, Y. Yu, Tapping on the potential of q&a community by recommending answer providers, in: Proceedings of the 17th ACM conference on Information and knowledge management, 2008, pp. 921–930.

[16] D.-R. Liu, Y.-H. Chen, W.-C. Kao, H.-W. Wang, Integrating expert profile, reputation and link analysis for expert finding in question-answering websites, Inf. Process. Manage. 49 (1) (2013) 312–329.

[17] T.P. Sahu, N.K. Nagwani, S. Verma, Taglda based user persona model to identify topical experts for newly posted questions in community question answering sites, Int. J. Appl. Eng. Res. 11 (10) (2016) 7072–7078.

[18] Z. Dai, C. Xiong, J. Callan, Z. Liu, Convolutional neural networks for soft-matching n-grams in ad-hoc search, in: Proceedings of the eleventh ACM international conference on web search and data mining, 2018, pp. 126–134.

[19] P.-S. Huang, X. He, J. Gao, L. Deng, A. Acero, L. Heck, Learning deep structured semantic models for web search using clickthrough data, in: Proceedings of the 22nd ACM international conference on Information & Knowledge Management, 2013, pp. 2333–2338.

[20] B. Mitra, F. Diaz, N. Craswell, Learning to match using local and distributed representations of text for web search, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 1291–1299.

[21] L. Yang, Q. Ai, J. Guo, W.B. Croft, anmm: Ranking short answer texts with attention-based neural matching model, in: Proceedings of the 25th ACM international on conference on information and knowledge management, 2016, pp. 287–296.

[22] A. Figueroa, G. Neumann, Learning to rank effective paraphrases from query logs for community question answering, in: Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, AAAI Press, 2013, pp. 1099–1105.

[23] D.H. Dalip, M.A. Gonçalves, M. Cristo, P. Calado, Exploiting user feedback to learn to rank answers in q&a forums: a case study with stack overflow, in: Proceedings of the 36th international ACM SIGIR conference on Research and development in information retrieval, 2013, pp. 543–552.

[24] G. Burel, P. Mulholland, Y. He, H. Alani, Predicting answering behaviour in online question answering communities, in: Proceedings of the 26th ACM Conference on Hypertext & Social Media, 2015, pp. 201–210.

[25] X. Cheng, S. Zhu, G. Chen, S. Su, Exploiting user feedback for expert finding in community question answering, in: 2015 IEEE International Conference on Data Mining Workshop (ICDMW), IEEE, 2015, pp. 295–302.

[26] S. Romeo, G. Da San Martino, A. Barrón-Cedeno, A. Moschitti, Y. Belinkov, W.-N. Hsu, Y. Zhang, M. Mohtarami, J. Glass, Neural attention for learning to rank questions in community question answering, in: Proceedings of COLING 2016, the 26th International Conference on Computational Linguistics: Technical Papers, 2016, pp. 1734–1745.

[27] J. Lee, S. Yun, H. Kim, M. Ko, J. Kang, Ranking paragraphs for improving answer recall in open-domain question answering, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 565–569.

[28] M. Dehghan, H.A. Rahmani, A.A. Abin, V.-V. Vu, Mining shape of expertise: A novel approach based on convolutional neural network, Inf. Process. Manage. 57 (4) (2020) 102239.

[29] T. Qin, T.-Y. Liu, J. Xu, H. Li, Letor: A benchmark collection for research on learning to rank for information retrieval, Inf. Retrieval 13 (4) (2010) 346–374.

[30] M. Asaduzzaman, A.S. Mashiyat, C.K. Roy, K.A. Schneider, Answering questions about unanswered questions of stack overflow, in: 2013 10th Working Conference on Mining Software Repositories (MSR), IEEE, 2013, pp. 97–100.

[31] L. Nie, Y. Li, F. Feng, X. Song, M. Wang, Y. Wang, Large-scale question tagging via joint question-topic embedding learning, ACM Transactions on Information Systems (TOIS) 38 (2) (2020) 1–23.

[32] G. Burel, Y. He, H. Alani, Automatic identification of best answers in online enquiry communities, Extended Semantic Web Conference, Springer (2012) 514–529.

[33] M. Kusner, Y. Sun, N. Kolkin, K. Weinberger, From word embeddings to document distances, in: International conference on machine learning, 2015, pp. 957–966..

[34] M. Surdeanu, M. Ciaramita, H. Zaragoza, Learning to rank answers to non-factoid questions from web collections, Comput. Linguist. 37 (2) (2011) 351–383.
[35] R. Senter, E.A. Smith, Automated readability index, CINCINNATI UNIV OH, Tech. rep., 1967.
[36] M. Coleman, T.L. Liau, A computer readability formula designed for machine scoring, J. Appl. Psychol. 60 (2) (1975) 283.
[37] J.P. Kincaid, R.P. Fishburne Jr, R.L. Rogers, B.S. Chissom, Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for navy enlisted personnel (Tech. rep.), Naval Technical Training Command Millington TN Research Branch, 1975.
[38] R. Flesch, A new readability yardstick, J. Appl. Psychol. 32 (3) (1948) 221–233.
[39] R. Gunning, The technique of clear writing, McGraw-Hill, New York, 1952.
[40] C.-H. Björnsson, Läsbarhet: Lesbarkeit durch Lix. (Aus dem Schwedischen), Liber (1968).
[41] G.H. Mc Laughlin, Smog grading-a new readability formula, J. Reading 12 (8) (1969) 639–646.
[42] D.M. Blei, A.Y. Ng, M.I. Jordan, Latent dirichlet allocation, J. Mach. Learn. Res. 3 (2003) 993–1022.
[43] M. Surdeanu, M. Ciaramita, H. Zaragoza, Learning to rank answers on large online qa collections, in: Proceedings of ACL-08: HLT, 2008, pp. 719–727..
[44] D. Kundu, R.K. Pal, D.P. Mandal, Time-aware hybrid expertise retrieval system in community question answering services, Appl. Intell. (2021) 1–18.
[45] A. Grover, J. Leskovec, node2vec: Scalable feature learning for networks, in: Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining, 2016, pp. 855–864.
[46] S.A. Alkhodair, B.C. Fung, O. Rahman, P.C. Hung, Improving interpretations of topic modeling in microblogs, J. Assoc. Inf. Sci. Technol. 69 (4) (2018) 528–540.
[47] T. Mikolov, E. Grave, P. Bojanowski, C. Puhrsch, A. Joulin, Advances in pre-training distributed word representations, in: Proceedings of the International Conference on Language Resources and Evaluation (LREC 2018), 2018.
[48] V. Bhat, A. Gokhale, R. Jadhav, J. Pudipeddi, L. Akoglu, Min(e)d your tags: Analysis of question response time in stackoverflow, in: 2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2014), 2014, pp. 328–335.
[49] E. Meij, W. Weerkamp, M. De Rijke, Adding semantics to microblog posts, in: Proceedings of the fifth ACM international conference on Web search and data mining, 2012, pp. 563–572.
[50] M. Choetkiertikul, D. Avery, H.K. Dam, T. Tran, A. Ghose, Who will answer my question on stack overflow?, 2015 24th Australasian Software Engineering Conference, IEEE (2015) 155–164