Learning to Leverage Microblog Information for QA Retrieval

Jose Herrera[1(✉)], Barbara Poblete[1], and Denis Parra[2]

[1] Department of Computer Science, University of Chile, Santiago, Chile
{jherrera,bpoblete}@dcc.uchile.cl
[2] Department of Computer Science, Pontificia Universidad Católica de Chile, Santiago, Chile
dparra@ing.puc.cl

Abstract. Community Question Answering (cQA) sites have emerged as platforms designed specifically for the exchange of questions and answers among users. Although users tend to find good quality answers in cQA sites, they also engage in a significant volume of QA interactions in other platforms, such as microblog networking sites. This in part is explained because microblog platforms contain up-to-date information on current events, provide rapid information propagation, and have social trust.

Despite the potential of microblog platforms, such as Twitter, for automatic QA retrieval, how to leverage them for this task is not clear. There are unique characteristics that differentiate Twitter from traditional cQA platforms (e.g., short message length, low quality and noisy information), which do not allow to directly apply prior findings in the area. In this work, we address this problem by studying: (1) the feasibility of Twitter as a QA platform and (2) the discriminating features that identify relevant answers to a particular query. In particular, we create a document model at conversation-thread level, which enables us to aggregate microblog information, and set up a learning-to-rank framework, using factoid QA as a proxy task. Our experimental results show microblog data can indeed be used to perform QA retrieval effectively. We identify domain-specific features and combinations of those features that better account for improving QA ranking, achieving a MRR of 0.7795 (improving 62% over our baseline method). In addition, we provide evidence that our method allows to retrieve complex answers to non-factoid questions.

Keywords: Twitter · Question Answering · Learning-to-rank

J. Herrera and B. Poblete have been partially funded by the Millennium Nucleus Center for Semantic Web Research under Grant NC120004. J. Herrera is partially funded by the CONICYT Doctoral Program and D. Parra is funded by FONDECYT under grant 2015/11150783. J. Herrera, B. Poblete and D. Parra have been partially funded by FONDEF under grant ID16I10222.

G. Pasi et al. (Eds.): ECIR 2018, LNCS 10772, pp. 507–520, 2018.
https://doi.org/10.1007/978-3-319-76941-7_38

1 Introduction

Online social networking platforms are designed to facilitate user interaction through the creation of diverse user communities. In particular, community Question Answering (cQA) websites are social platforms that specialize in connecting users that have questions (i.e., information needs expressed as questions in natural language) with other users that can provide answers to them. Examples of these platforms are Y! Answers[1], Stack Exchange[2], among others.

User exchanges in cQA web sites are commonly preserved online indefinitely, becoming in time historic knowledge bases. Once consolidated, these knowledge bases become rich repositories for finding answers to newly formulated questions in the platform, or to queries in search engines, which have been phrased in the form of questions. These resources are important for information retrieval (IR) tasks related to question answering (QA), because questions commonly convey complex information needs, which are difficult to satisfy using traditional IR techniques [16,20].

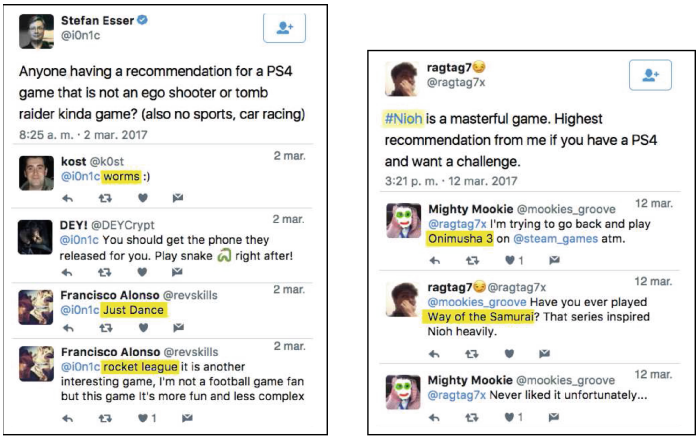
Despite the significance of cQA platforms, these are not the only way in which users ask and answer questions. Prior work shows that users of the microblog social networking platform Twitter[3], also engage in an important amount of QA, accounting for almost 10% its activity [8,17,26]. For example, Fig. 1 shows two Twitter conversation threads related to the question "Which games are good for ps4?". In this example each conversation provides several relevant answers, which can all be considered as part of the answer to the original question.

Twitter, in particular, has a large user-base[4] and allows users to quickly exchange vast amounts of information through short messages (called *tweets*). Users commonly use Twitter to post real-time status updates, share information on diverse topics and chat with other users. Researchers believe that the immediacy of information on Twitter is one driver for users to engage in QA in this platform, but also because users seek answers from their preferred social network [17,19]. This indicates that the answers sought by Twitter users probably have a temporal, social and/or geographical context, which can be quickly and easily addressed by other users in this network.

The sustained use of Twitter for QA suggests that valuable historic information for automatic QA retrieval could be obtained from this platform. Furthermore, this information could complement that provided by traditional cQA web sites, with more up-to-date and context rich answers.

In this work we addressed the problem of how to effectively leverage Twitter data for QA retrieval. In particular, as an initial approach on this topic, we performed an investigation based on the following three research questions:

[1] http://answers.yahoo.com.
[2] http://stackexchange.com.
[3] http://www.twitter.com.
[4] 328 million users in June 2016 (https://about.twitter.com/es/company).



(a) Conversation thread 1. (b) Conversation thread 2.

Fig. 1. Example of two Twitter conversation threads that answer the question: *"Which games are good for ps4?"*. User suggestions are highlighted in yellow. (Color figure online)

RQ1: Is it possible to retrieve relevant answers to incoming questions using historic Twitter data?

RQ2: Which features are the most important for finding relevant answers to questions using Twitter data?

RQ3: Can Twitter conversation threads be used to answer questions (as opposed to using single tweets) in order to provide answers to complex questions?

We undertook this task by introducing a novel document representation, which considers complete conversation threads as documents. Using a learning-to-rank (LTR) framework we studied which were the most important features for re-ranking relevant conversation threads (referred to as *threads* from now on). We learned and evaluated this approach by using standard factoid QA datasets (i.e., on questions that can be answered by a simple fact). We also performed a manual analysis of the applicability of our approach on non-factoid questions.

We found that, in general, by using a thread-level document representation of microblog data, we are in fact able to re-rank relevant answers in the top positions. Our experiments show that the best single feature set for ranking is parts-of-speech (POS). Furthermore, by combining POS with other features such as: distance-based, social-based, and word embedding features, we are able to achieve a MRR of 0.7795. Improving 18% over the best single performing feature (POS) and 62% over the baseline (BM25). In addition, our manual evaluation of non-factoid questions indicates that our approach is also a very good starting point for answering much more complex and context rich questions.

This document is organized as follows: Sect. 2 discusses relevant prior work in the area, Sect. 3 describes our problem statement and proposed solution.

Section 4 presents our experimental setup, as well as results and discussion, and Sect. 5 conclusions and ideas for future work.

2 Related Work

Previous investigations in QA have identified relevant features for cQA retrieval. However, the particular characteristics of microblog platforms, makes the problem of QA retrieval notably different than that addressed in prior work. For instance, an important difference between microblog and cQA platforms is the length of messages, which requires document models and features that specifically target short text [21]. Another difference, is that users on cQA platforms tend to be more specialized, focusing on one, or very few, topics of interest [9], unlike the Twitter community which tends to be miscellaneous.

QA in non-cQA platforms. There have been studies that analyze how users ask, and what they ask, in non-specific QA social networks. Morris et al. [17] performed a characterization study of questions asked in Facebook and Twitter, and found that the most asked questions are about recommendations, opinions and factual knowledge. Paul et al. [19] found that the most popular questions in Twitter are rhetorical and of factual knowledge. In addition, they observed that roughly 18.7% of the questions received at least one reply; the first within 5–30 min and the remainder within the next 10 h. Zhao et al. [26] extracted features from tweets and built a classifier that distinguishes real questions (i.e., questions that seek answers) from rhetorical questions. In addition, Liu et al. [12] created a taxonomy to describe the types of questions that user ask in Twitter.

LTR and features for QA retrieval. Learning-to-rank (LTR) refers to machine learning techniques for training a model for a ranking task. LTR is widely used in several types of ranking problems in information retrieval (including traditional QA), natural language processing and recommender systems; For example, Duan et al. [3] used LTR to rank tweets according to how informative they were, based on content features such as URLs and length. Molino et al. [15] used LTR to find the *best answers* in Yahoo Answers and Surdenau et al. [22] studied non-factoid questions in the same platform, exploring feature combinations.

We complement prior work by analyzing which features contribute to find relevant answers in Twitter and how they differ from those of cQA platfomrs. In this article we extend our short workshop paper Herrera et al. [7], which introduces the initial problem of using microblogs for QA retrieval. This extended version has been completely rewritten and improved by adding: (1) a formal problem statement, (2) an experimental framework for ranking Twitter conversations for QA retrieval, (3) an experimental evaluation of its effectiveness, and (4) by analyzing the features which contribute most.

3 Microblog Ranking for QA Retrieval

In this section we present our approach for leveraging microblog information for QA retrieval tasks. We model our research problem as that of *re-ranking*,

in which the main goal is to rank relevant answers in the top result positions. For our current purpose, we define an answer as *relevant* if it contains a correct answer to the original question. Then, we study which microblog features have the most influence for determining relevant answers. Specifically, we propose an aggregated *thread-level document model*, which considers conversation *threads* as documents for retrieval. This representation allows us to use aggregated information as documents for answering queries, as opposed to using a single tweets. Our hypothesis is that the composition of several tweets into a single thread can provide answers to complex questions.

3.1 Problem Statement for Microblog QA

More formally, let q^* be a question that corresponds to an information need formulated by a user. Let $Q^* = \{q_1, q_2, \ldots, q_n\}$ be the set of possible query formulations of q^*. We define query formulations in Q^* as any variation of the initial input query q^* that allows us to retrieve a set of threads that are candidate answers for q^* [10]. Then, for each $q_i \in Q^*$, we extract all of the threads (documents) that match q_i in a given microblog dataset. In particular, we say that a thread t_i matches query q_i when t_i contains *all of the terms* in q_i. Next, let $\mathcal{T} = \{t_1, t_2, \ldots, t_m\}$ be the set that contains the union of the sets of threads that match the query formulations in Q^*, and therefore by extension, which match q^*.

Hence, the goal of our research is, for a given question q^*, to re-rank the threads in $\mathcal{T}$ according to their relevance for answering q^*. We define this as a re-ranking problem since our starting point are the set of threads in $\mathcal{T}$, which are initially obtained using a very simple retrieval method (detailed in Sect. 4). In other words, our goal is to learn a function $f(q^*, \mathcal{T}) \rightarrow \pi$ that produces an optimal permutation $\hat{\pi}$ of the elements in $\mathcal{T}$ for answering the question q^*.

We acknowledge that there are other important tasks besides ranking, in QA retrieval such as, creating the best possible query formulations and selecting the passages within a text that contain the answer to a question [10]. However, at this moment we consider those problems as beyond the current scope of our work.

3.2 Proposed Solution

We propose a solution based on a LTR framework that will allow us to learn the aforementioned function $f(q^*, \mathcal{T})$. In order to identify the features that produce the best ranking, we evaluate several combinations of sets of features using different LTR models. In particular we use the following four LTR models, defined next: MART [6], Ranknet [1], Rankboost [5] and LambdaMart [24].

This solution also requires us to specify: (1) the *query formulations*, (2) the *features* that will be used:

(1) Query formulations. Since our focus is on re-ranking, we select a set of broad query formulations that increase the recall of documents that may contain an answer to the query q^*. In particular, we use the following query formulations:

- **q_1**: Corresponds to the original question as it was formulated by the user q^*.
- **q_2**: Corresponds to q^* after lowercase and whitespace normalization, removal of non-alphanumerical characters and terms with only one character.
- **q_3**: Corresponds to q_2 after the additional removal of stopwords, with the exception of terms in the **6W1H**[5]. For example, the question q^* = "What is the scientific name of tobacco?" becomes q_3 = "what scientific name tobacco".
- **q_4**: Corresponds to q_3 without the **6WH1**. In the previous example, q^* would be transformed to q_4 ="scientific name tobacco".

(2) Features[6]. We performed a review of the features used in prior work for traditional QA retrieval and adapted those that could be applied to microblog data. In addition, we manually inspected approximately 1,000 QA threads in Twitter (i.e., threads in which their initial tweet is a question, shown in Fig. 1). This inspection allowed us to identify features that could potentially help determine whether the question in the QA thread was answered inside the thread itself or not. We describe the different types of features next:

- **Distance-based features (D_TFIDF_N, D_WEMB):** These features are based on four well-known distance metrics, between the query q^* and a thread t. *cosine, manhattan, euclidean and jaccard* [2]. We compute these distances using 2 types of vector representations: (i) **D_TFIDF_N**, which are the aforementioned distances between the *tf-idf* vector representations of q^* and t (using n-grams of size $N = \{1, 2, 3\}$), and (ii) **D_WEMB**, which are the same distances but using the word embedding vector representations of q^* and t.
- **Social-based features (SOCIAL):** These features are based on the social interactions observed in a conversation threads (i.e., thread level features). These include: number of replies in a thread, number of different users that participate, fraction of tweets with favorites/retweets/hashtags, number of user mentions, and number of different user mentions.
- **User-based features (USER):** These features are based on properties to the users that participate in a conversation thread. These include: total number of followers and followees of the users that participate in a thread, the fraction of users in the thread that have a verified account, the average *age* of the users in a thread. User age is computed as the difference between date of creation of the Twitter user account and the date when the tweet was posted.
- **Content-based features (CONTENT):** These features are based on the content of a thread. These include: the number of different URLs in the thread, the number of words (removing URLs and punctuation), the length of the thread $\frac{\#\ words}{\#\ tweets}$ (considering only words with size ≥ 1), the fraction

[5] **6W1H** corresponds to 5WH1 with the addition of the terms "Which" (i.e. Who, What, Where, When, Why, Which and How).

[6] Due to space constrains only a high-level description of the features is provided. However, the detailed list of features is available at https://goo.gl/qqACz5.

of uppercase and lowercase letters, the number of positive/negative/neutral emoticons, and the average number of words in English. Some social-based, user-based and content-based features have been adapted to microblog data from features used in prior works [3,15].

- **Part-of-speech (POS) features:** These features are based on part-of-speech tagging. In particular we compute the frequency of each high-confidence POS tag in a conversation thread, using the Twitter-specific tagger *TweetNLP* by [18].
- **Representative words feature (REPW):** This feature corresponds to the fraction of *"representative words"* that are contained in a thread. Where a "representative word" is any word which is contained in the top-50% most frequent terms over all threads in the training data (excluding stopwords).
- **Word embedding based features (WEMB_THR, WEMB_Q):** This feature is computed using the explicit vector representation of the query (WEMB_Q) and the thread (WEMB_THR), created using Word2Vec [14] and a pre-trained model on 400 million tweets[7]. Each vector was composed by 300-dimensions and the word embeddings are inferred using a skip-gram architecture.
- **Time-based features (TIME):** These features include time-based characteristics of the thread, such as: time-lapse between the first tweet in the thread and the last, and the average time between tweets in a thread.

4 Experimental Evaluation and Results

Our main goal is to validate the feasibility of retrieving and ranking relevant answers to questions using past information exchanges on Twitter (**RQ1**). In addition, we also want to measure the contribution of the different microblog features for ranking (**RQ2**), and to see if a thread-level document representation can help provide answers to non-factoid questions (**RQ3**). We focus on two types of evaluation, *factoid QA task* and *non-factoid QA task*. The first is a quantitative evaluation based on standard QA ground truth datasets. The second evaluation is more exploratory and is based on a manual evaluation.

4.1 Factoid QA Evaluation Task

We evaluate our approach on factoid questions because there are several benchmark QA datasets available to validate automatically correct answers. Jurafsky and Martin [10] defined as *factoid* question answering those tasks that require one answer, which is a simple fact (e.g.,"Who was the first American in space?"). Nevertheless, we use the factoid task as a proxy to our goal of answering more complex questions (i.e., non-factoid).

[7] http://www.fredericgodin.com/software/.

Ground-truth dataset. We built a ground truth QA dataset based on datasets provided by the TREC QA challenge[8] (*TREC-8 (1999)*, *TREC-9 (2000)*, *TREC-2004* and *TREC-2005*) and a repository of factoid-curated questions for benchmarking Question Answering systems[9]. All of these datasets, except for TREC-8, provided regular expressions to match correct answers automatically. For the remaining dataset, we manually created regular expressions to match the correct answers. In addition, we manually removed questions that were: time-sensitive (e.g.,"What is the population of the Bahamas?"), inaccurate (e.g.,"what is the size of Argentina?"), not phrased as a question (e.g.,"define thalassemia"), referred to other questions (e.g., "What books did she write?") and questions whose length were over 140 characters. This resulted in a ground-truth dataset of 1,051 factoid questions.

Twitter dataset for LTR factoid questions. In an ideal scenario, our candidate answers would be obtained from a large historical Twitter dataset, or from the complete data-stream. However these types of data repositories were not available to us for this evaluation. We then approximated the ideal retrieval scenario by using the Twitter Search API [10] as an endpoint, which provides access to a sample of the actual data. For each query formulation, described in Sect. 3.2, we retrieved as many tweets as possible up to 1,000 tweets per query. In addition, if the retrieved tweet was part of a conversation thread we also retrieved that thread in full. Overall, we found candidate answers for 491 (47%) of the questions (i.e., the remaining questions had no matching threads, e.g. "What was the ball game of ancient Mayans called?"). To improve the learning process, we removed low relevance threads (i.e., the cosine similarity between thread and query was ≤ 0.3) and threads that had no replies. The resulting dataset contained 33,873 conversation threads with 63,646 tweets. We note that our datasets (ground-truth and Twitter) are publicly available[11].

Baseline Methods. We compare our approach to the following methods, which have been used as baselines in prior cQA studies [15,22,23]:

- **BM25:** The Okapi weighting BM25 is widely used for ranking and searching tasks [13]. We use the BM25 document score in relation to a query. We use $b = 0.75$ and $k_1 = 1.2$ as parameters since they were reported optimal for other IR collections [23].
- **Twitter Search:** This method lists results from Twitter's search interface. Results are obtained by searching for each query in Q^* using the "latest" option, which lists messages from the most recent to the oldest message. The results obtained for each query are then joined in chronological order. However, this method is not reproducible since it works like a "black box" from our perspective.

[8] http://trec.nist.gov/data/qamain.html.
[9] https://github.com/brmson/dataset-factoid-curated.
[10] https://dev.twitter.com/rest/public/search.
[11] https://github.com/jotixh/ConversationThreadsTwitter/.

– **REPW:** This method uses the feature REPW, described in Sect. 3, with the best performing LTR model from our experiments. Experimentally, this method behaves as an upper bound of the "Twitter search" method with the advantage that it can be reproduced.

Evaluation methodology. We built several models that rank tweets and conversation threads as potential answers to a set of given factoid questions. We use the LTR software library Ranklib of the LEMUR project[12] for this task. To reduce the probability of obtaining significant differences among the LTR methods only by chance, we relied on bootstrapping [4]. Rather than having a single train/test split of the dataset or cross-validation, which did not work well in practice, we sampled with replacement 30 random collections. Then, each collection was divided into 70% of the questions for training (with their respective tweets/threads) and 30% for testing.

We evaluated different combinations of sets of features in every experiment. For each of these combinations we computed MRR@10 and $nDCG$@10. In each case we report the mean value over the 30 bootstrapped collections. We ran the experiments using the default *Ranklib* parameters for the LTR methods used: MART, Ranknet, RankBoost, and LambaMART.

Feature set selection. We propose the following heuristic to find the best combination of features, where f_i is a feature set (e.g. POS features), $F = \{f_1, f_2, \ldots f_n\}$ is the set that contains all of our feature sets, and PBC (initially, $PBC = \emptyset$) is the partial best feature set combination:

1. We run the factoid task evaluation for each feature set in F using each LTR model.
2. We choose the feature set f_i^* which produces the best MRR@10 and add it to the set PBC (i.e., $PBC = PBC \cup f_i^*$) and we remove f_i^* from F (i.e. $F = F - f_i^*$).
3. We run again the same evaluation using the resulting PBC in combination with each remaining feature in F (i.e., $(PBC \cup f_1), (PBC \cup f_2) \ldots (PBC \cup f_n)$).
4. We repeat the process from the step (2) until there is no significant improvement in the MRR@10 value.

Single feature results. Table 1 presents the results of each LTR model trained on different types of features. These results show that features obtained from the text of the messages (POS, WEMB_THR, CONTENT) yield good results compared to, for instance, relying solely on social signals such as replies, likes or retweets (SOCIAL). The single most predictive feature set for ranking answers to factoid questions is *Part-of-Speech* (POS), which significantly outperforms all the other features.

Feature combination results. Table 2 shows the results of several experiments combining feature sets in the LTR framework. The table shows the percent of improvement over the best performing feature set POS (MRR@10 = 0.6587), and

[12] https://sourceforge.net/p/lemur/wiki/RankLib/.

Table 1. Factoid task MRR@10 results, mean (μ) and S.D. (σ). POS($^{2-12}$) means that POS is significantly better than feature sets 2 (WEMB_THR) to 12 (REPW).

	Feature set	MART		Ranknet		RankBoost		LambdaMart	
		μ	σ	μ	σ	μ	σ	μ	σ
1	POS ($^{2-12}$)	**0.6587**	0.0250	0.5862	0.0266	**0.6730**	0.0377	0.6213	0.0617
2	WEMB_THR ($^{3-12}$)	**0.6202**	0.0296	0.5489	0.0315	0.6013	0.0264	0.5618	0.0388
3	CONTENT ($^{4-12}$)	0.5763	0.0284	0.5694	0.0320	0.5543	0.0230	0.5900	0.0330
4	D_TFIDF_1 ($^{9-12}$)	0.5282	0.0286	0.5299	0.0349	0.5143	0.0311	0.4966	0.0407
5	SOCIAL ($^{8-12}$)	0.5280	0.0284	0.5490	0.0265	0.4766	0.0296	0.5311	0.0424
6	D_WEMB (^{9,11,12})	0.5131	0.0303	0.5278	0.0313	0.5155	0.0337	0.5105	0.0341
7	D_TFIDF_3 (^{9,11,12})	0.5123	0.0331	0.4057	0.0262	0.4716	0.0353	0.4075	0.0280
8	D_TFIDF_2 (^{9,11,12})	0.5083	0.0303	0.4457	0.0277	0.4870	0.0315	0.4338	0.0254
9	USERS	0.4857	0.0223	0.5344	0.0296	0.4883	0.0278	0.5376	0.0503
10	WEMB_Q	0.4942	0.0258	0.4942	0.0258	0.4942	0.0258	0.4942	0.0258
11	TIME	0.4815	0.0428	0.5150	0.0303	0.4942	0.0258	0.5560	0.0315
12	REPW	0.4810	0.0326	0.3651	0.0347	0.4929	0.0328	0.4051	0.0677

Significant differences based on MART pairwise t-tests, $\alpha = .95$, Bonferroni correction.

Table 2. Factoid task, best combinations of features sets, based on MRR@10, and their percent of improvement over the best single feature set (POS).

Combination	MART	Ranknet	RankBoost	LambdaMart
POS	0.6587	0.5862	0.6730	0.6213
POS+D_TFIDF_1	0.6917 ↑ 5%	0.5953	0.6746	0.6200
POS+D_TFIDF_1+SOCIAL	0.7514 ↑ 14%	0.5931	0.6719	0.6361
POS+D_TFIDF_1 +SOCIAL+WEMB_Q	0.7682 ↑ 17%	0.5946	0.6719	0.6464
POS+D_TFIDF_1+SOCIAL+ WEMB_Q+D_TFIDF_3	0.7745 ↑ 18%	0.5904	0.6732	0.6204
POS+D_TFIDF_1+SOCIAL+ WEMB_Q+D_TFIDF_3+REPW	0.7788 ↑ 18%	0.5895	0.6733	0.6415
POS+D_TFIDF_1+SOCIAL+ WEMB_Q+D_TFIDF_3+REPW+TIME	**0.7795** ↑ 18%	0.5867	0.6755	0.6420

we show that a combination with content (D_TFIDF_1, WEMB_Q, D_TFIDF_3, REPW), social and time feature sets can increase the performance up to 18.3% (MRR@10 = 0.7795), showing that these features provide different types of signals for the ranking task.

Methods. Considering both evaluations –on each feature set and over combinations– the best method was MART, specially in the feature set combination results of Table 2. Although LambdaMart is usually presented as the state of the art, there is also recent evidence on non-factoid QA showing MART as the top performing algorithm [25], in line with our results. Notably, all the methods show a strongly correlated behavior in terms of feature set ranking, for the three of them present their best MRR@10 results with the POS feature and their worst

Table 3. Left: Results of our best combination vs. baselines. We improve over Twitter search up to 74.77% (MRR@10) and 29.4% (nDCG@10). **Right:** Datasets description of non-factoid and factoid QA. Size differences justify the need for transfer learning.

Method	MRR@10	nDCG@10
BM25	0.3852	0.4793
Twitter Search	0.4460	0.5625
REPW	0.4810	0.4616
Best comb.	**0.7795**	**0.7279**

	Non-fact	Fact
# of questions	40	491
# of tweets	2,666	63,646
# of threads	386	33,873
% tweets that are part of a thread	87.99%	46,7%
Avg. replies per thread	3.32	0.9

results with the REPW feature (with the exception of RankBoost), as shown in Table 1. This consistent behavior underpins our conclusions in terms of the importance of POS for this task.

Baselines. Results in Table 3 (left). Our best combined LTR model beats all baselines, improving the factoid ranking results by 74.77% in terms of MRR@10 and by 29.4% on nDCG@10 over Twitter search.

4.2 Non-factoid QA Evaluation Task

Non-factoid questions can have more than one answer, and they are usually associated to questions that require opinions, recommendations, or communicate experiences. For example, some non-factoid questions found in Twitter are: "anyone have any GIF maker software or sites they can recommend?" and "Anyone have a remedy for curing a headache?". To perform a preliminary evaluation of our approach for this task, we sampled 40 diverse non-factoid questions from Twitter. We focused on these questions because they represent an important portion (30%) [17] of the questions asked by users and they can be retrieved in a simple way (i.e., we retrieved these questions searching for "recommend ?"). We then obtained candidate answers for each question in the same way as in our factoid task (Sect. 4.1). Next, we used transfer learning (i.e., we use our best factoid QA LTR model) to rank answers for the non-factoid task. The differences in sizes of our datasets, shown in Table 3 (right), justify transferring our existing model, rather than learning a new one from non-factoid data.

Unlike the TREC dataset of factoid questions, we do not have the ground truth of correct answers. Hence, we manually inspected and evaluated the top-15 answers ranked with our approach for each of the 40 non-factoid questions, labeling them as relevant and non-relevant. Table 3 (right) shows the characteristics of both the factoid and non-factoid datasets.

Results. We obtained a $MRR@10 = 0.5802$, which is good compared to results reported recently –MRR = [0.4−0.45] in [25]–, but suboptimal compared to what we obtained in factoid QA task. By further analyzing the data we found that, in average, for every question we retrieved 1.5 threads without any reply, which were also non relevant to the question made. Based on this, we discarded from potential answers those threads without replies (or single tweets). This strategy

improved results, $MRR@10 = 0.6675$, with the small trade-off of one question out of 40 for which we could not find answers.

Discussion and Limitations. An interesting result of our evaluation was the high predictive power of part-of-speech features. Previous work on community QA conducted on a large dataset of Yahoo Answers [15] found similar results. We also found a good discriminative effect of coordinating conjunctions (and, but, or, so). Hence, stopwords should not be removed for POS tagging in the factoid QA task using microblog data. However, we also found differences with prior findings on cQA [15], which observed that punctuation was a discriminative feature between relevant and non-relevant answers, unlike our study of QA in microblogs. This is expected since the short nature of microblog text makes people less likely to use punctuation. In addition, Molino et al. [15] used a feature similar to D_WEMB (i.e., the distance between the query and the candidate answers based on word2vec), but this neural-based word embedding feature did not perform well, ranking in 30th place among other features. In our evaluation, we used distances but also the word embeddings directly as features, which yielded excellent results, ranking as the 2nd most important feature set. This indicates that for microblog QA it is better to use the values of the embedded dimensions as features. Our manual inspection of results indicates that transfer learning can be a potential way to perform non-factoid QA, by using a model pre-trained for factoid QA.

A limitation in our work is that, for factoid QA, we could only find answers for about 40% of the questions in our ground-truth dataset. We note our initial factoid dataset, based on TREC challenges, does not have topics related to current events, which are much more likely to be discussed in Twitter [11]. This time gap between our ground-truth questions and our candidate answers, can very likely explain why we were unable to find matching tweets for an important number of questions. Another limitation, which we plan to address in the future, is that we did not study the occurrence of *incorrect answers* within relevant threads.

5 Conclusion and Future Work

In this work we investigated the feasibility of conducting QA using microblog data. We studied several sets of features, ranking methods and we performed a quantitative evaluation on a factoid QA dataset, as well as an informative evaluation with non-factoid questions. Our results validate the potential for using microblog data for factoid and non-factoid QA, identifying the most informative features as well as the best LTR model. In future work we expect to conduct a larger evaluation on non-factoid questions, using new features, and performing a deeper analysis on the effect of certain attributes.

References

1. Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., Hullender, G.: Learning to rank using gradient descent. In: Proceedings of ICML 2005, pp. 89–96 (2005)
2. Büttcher, S., Clarke, C.L.A., Cormack, G.V.: Information Retrieval -Implementing and Evaluating Search Engines. MIT Press, Cambridge (2010)
3. Duan, Y., Jiang, L., Qin, T., Zhou, M., Shum, H.Y.: An empirical study on learning to rank of Tweets. In: Proceedings of COLING 2010, pp. 295–303 (2010)
4. Efron, B., Tibshirani, R.J.: An Introduction to the Bootstrap. CRC Press, Boca Raton (1994)
5. Freund, Y., Iyer, R., Schapire, R.E., Singer, Y.: An efficient boosting algorithm for combining preferences. J. Mach. Learn. Res. **4**, 933–969 (2003)
6. Friedman, J.H.: Greedy function approximation: a gradient boosting machine. Ann. Stat. **29**(5), 1189–1232 (2001)
7. Herrera, J., Poblete, B., Parra, D.: Retrieving relevant conversations for Q&A on Twitter. In: Proceedings of ACM SIGIR (Workshop of SPS) (2015)
8. Honey, C., Herring, S.C.: Beyond microblogging: conversation and collaboration via Twitter. In: Proceedings of HICSS 2009, pp. 1–10 (2009)
9. Java, A., Song, X., Finin, T., Tseng, B.: Why we Twitter: understanding microblogging usage and communities. In: Proceedings of WebKDD/SNA-KDD 2007, pp. 56–65 (2007)
10. Jurafsky, D., Martin, J.H.: Speech and Language Processing. Prentice Hall, Pearson Education International (2014)
11. Kwak, H., Lee, C., Park, H., Moon, S.: What is Twitter, a social network or a news media? In: Proceedings of WWW 2010, pp. 591–600 (2010)
12. Liu, Z., Jansen, B.J.: A taxonomy for classifying questions asked in social question and answering. In: Proceedings of CHI EA 2015, pp. 1947–1952 (2015)
13. Manning, C.D., Raghavan, P., Schütze, H.: Introduction to Information Retrieval. Cambridge University Press, Cambridge (2008)
14. Mikolov, T., Chen, K., Corrado, G., Dean, J.: Efficient estimation of word representations in vector space. In: Proceedings of ICLR (2013)
15. Molino, P., Aiello, L.M., Lops, P.: Social question answering: textual, user, and network features for best answer prediction. ACM TOIS **35**, 4–40 (2016)
16. Morris, M.R., Teevan, J., Panovich, K.: A comparison of information seeking using search engines and social networks. In: Proceedings of ICWSM 2010, pp. 23–26 (2010)
17. Morris, M.R., Teevan, J., Panovich, K.: What do people ask their social networks, and why?: a survey study of status message Q&A behavior. In: Proceedings of CWSM 2010, pp. 1739–1748 (2010)
18. Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., Smith, N.A.: Improved part-of-speech tagging for online conversational text with word clusters. In: Proceedings of ACL (2008)
19. Paul, S.A., Hong, L., Chi, E.H.: Is Twitter a good place for asking questions? a characterization study. In: Proceedings of CWSM 2010, pp. 578–581 (2011)
20. Raban, D.R.: Self-presentation and the value of information in Q&A websites. JASIST **60**(12), 2465–2473 (2009)
21. Sriram, B.: Short text classification in Twitter to improve information filtering. In: Proceedings of ACM SIGIR 2010. ACM (2010)

22. Surdeanu, M., Ciaramita, M., Zaragoza, H.: Learning to rank answers on large online QA collections. In: Proceedings of ACL 2008, pp. 719–727 (2008)
23. Surdeanu, M., Ciaramita, M., Zaragoza, H.: Learning to rank answers to non-factoid questions from web collections. Comput. Linguist. **37**(2), 351–383 (2011)
24. Wu, Q., Burges, C.J.C., Svore, K.M., Gao, J.: Adapting boosting for information retrieval measures. Inf. Retrieval **13**(3), 254–270 (2010)
25. Yang, L., Ai, Q., Spina, D., Chen, R.C., Pang, L., Croft, W.B., Guo, J., Scholer, F.: Beyond factoid QA-effective methods for non-factoid answer sentence retrieval. In: Proceedings of ECIR (2016)
26. Zhao, Z., Mei, Q.: Questions about questions: an empirical analysis of information needs on Twitter. In: Proceedings of WWW 2013, pp. 1545–1556 (2013)