Social QA in non-CQA platforms

José Herrera[a], Denis Parra[b], Barbara Poblete[a]

[a]*Department of Computer Science, University of Chile*
[b]*Department of Computer Science, Pontificia Universidad Católica de Chile*
Santiago, Chile

Abstract

PRE-PRINT, to be published in Future Generation Computer Systems `https://doi.org/10.1016/j.future.2019.12.023`

Community Question Answering (cQA) sites have emerged as platforms designed specifically for the exchange of questions and answers among communities of users. Although users tend to find good quality answers in cQA sites, there is evidence that they also engage in a significant volume of QA in other types of social sites, such as microblog platforms. Research indicates that users opt for these non-specific QA social networks because they contain up-to-date information on current events, also due to their rapid information propagation, and social trust. In this sense, we propose that microblog platforms can emerge as a novel, valuable source of information for QA information retrieval tasks. However, we have found that it is not straightforward to transfer existing approaches for automatically retrieving relevant answers in traditional cQA platforms for use in microblogs. This occurs because there are unique characteristics that differentiate microblog data from that of traditional cQA, such as noise and very short text length. In this work, we study 1) if microblog data can be used to automatically provide relevant answers for the QA task, and, in addition, 2) which features contribute the most for finding relevant answers for a particular query. In particular, we introduce a conversation (thread) -level document model, as well as a machine learning ranking framework for microblog QA. We validate our proposal by using factoid-QA as a proxy task, showing that Twitter conversations can indeed be used to automatically provide relevant results for QA. We are able to identify the importance of different features that contribute the most for QA ranking. In addition, we provide evidence that our method allows us to retrieve complex answers in the domain of non-factoid questions.

Keywords:
Ranking, Question Answering, Relevance, Microblogs

1. Introduction

Online social networking platforms have changed how people produce and consume Web content. Social networking sites are designed to facilitate interaction among people, allowing for the creation of communities with different purposes. In particular, community Question Answering (cQA) Web sites are platforms that specialize in connecting users that are interested in expressing information needs in the form of questions, with other users that can provide answers to these questions. Examples of such sites are Yahoo! Answers[1], Stack Exchange[2], among others. Here, people can ask two kinds of questions: factoid and non-factoid questions. The first one requires just one answer or statement of fact, for example, *"What is the capital of France?"* The second one requires experiences, opinions, lists, recommendations or advice. For example, *"What are the best German beers?"*. The resulting interactions among users are traditionally preserved permanently in the cQA web site, constituting a historical knowledge base. This type of knowledge base can be very valuable, given that queries phrased as questions often represent complex information needs, which are not easily satisfied using traditional Web search engines [1, 2]

On the other hand, microblog platforms, such as Twitter[3], are general-purpose social networks (unlike cQA platforms) that allow users to exchange vast amounts of information through short messages (called *tweets*). In this sense, microblog users, publish real-time status updates and information about an unlimited number of diverse topics. These social networks have had large adoption worldwide, and are characterized for their ability to quickly disseminate information, in particular during high impact events. Microblog data stream at a fast pace, which can make content short lived and volatile for the average user, since it is continuously being replaced by newly arriving messages. A detailed look at Twitter user behavior has shown that its users are using this platform for QA related conversations on a regular ba-

[1] `http://answers.yahoo.com`
[2] `http://stackexchange.com`
[3] `http://www.twitter.com`

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

Figure 1: Example of two conversation threads extracted from Twitter that can answer the initial question: *"What are good games for ps4?"* (ps4: playstation 4). The initial tweets are paraphrased as questions followed by replies. User suggestions are highlighted in yellow.

sis [3, 4, 5], in which tweets with questions account for 10% of total messages. This use of microblogs can appear as counterintuitive, considering that there are other platforms specifically designed for QA.

However, this behavior can be explained by the fact that users are seeking answers from trusted sources (i.e., their day-to-day social network) [4, 6].

Figure 1 shows an example of two conversation threads from Twitter related to the question *"What are good games for ps4?"* The example shows that each thread provides several relevant answers for the question, which can only be identified by reviewing the complete conversation.

Another aspect that is taken into consideration by users is the immediacy of information provided by Twitter, in particular for current events and breaking news. Both of these drivers, implicitly imply that answers to users' questions, will have temporal, social and possibly geographical contexts. In addition, Twitter's real-time propagation properties can provide valuable information for emergency situations (e.g., earthquakes and floods). To illustrate this we study tweets related to the Paris terrorist attacks on November 15, 2015. The first message about the event was posted 3 min-

utes after the event (an explosion in a stadium) occurred: *"Explosion in the Stade de France? Was it a bomb or was it harmless? Explosion today here in France, in the stadium."*. After this, a burst of tweets arrived in the first 5-20 minutes, asking, commenting or expressing uncertainty about the event.

In this current work we focus on the problem of leveraging Twitter data for the task of QA retrieval. To achieve this we defined the following 3 research questions:

- **RQ1:** Is it possible to retrieve relevant answers to incoming questions using historic Twitter data?
- **RQ2:** Which types of features are important for finding relevant answers to questions using Twitter data?
- **RQ3:** How important is conversation context for identifying relevant answers?

To address these questions, we used as a ground truth a public benchmark dataset of factoid questions and their exact answers. For each question in the ground truth dataset, we retrieved from Twitter a set of recent conversation threads that constituted candidate answers for the question. Then, using learning-to-rank (LTR) methods, we trained a model to rank relevant Twitter answers for factoid queries. Based on this, we studied which were the most important features for re-ranking relevant conversations (i.e., conversations that contained correct answers to the question). Finally, we studied the effectiveness of this approach for the more general problem of answering *non-factoid questions* that users posted in Twitter. We did so, by constructing a set of questions posted by Twitter users requesting *recommendations* (see Figure 1). Our experimental findings indicate that automatic QA using microblog data is different to that of automatic QA using data from traditional cQA platforms. In particular, our experimental findings show that part-of-speech (POS) features are essential for accurately ranking relevant answers. Also important are social and user-based features, but in less degree than POS. Moreover, ranking considerably improves when combining POS features, distance-based features, social-based features, and word embeddings.

Overall, the main contributions presented in this article are:

1. To propose the use of social media posts in microblog platforms as a data source for automatic QA.

2. To introduce a framework for automatic QA retrieval based solely on Twitter data.
3. To perform an empirical evaluation, using publicly available datasets, of the effectiveness of using Twitter data as an additional source for automatic QA tasks.
4. To identify which microblog features contribute the most to finding relevant answers and to show that conversation context is important for this task.

Relation to previously published work: This article is an extension of our conference paper, Herrera et al. [7]. This extended version adds an in-depth description of our proposed QA model, as well as an expanded experimental analysis, which includes a novel characterization of replies and thread candidates in Twitter. Moreover, we have included an empirical study of more complex questions (non-factoid), with examples and a discussion.

2. Related Work

Q&A websites are a service where anyone can express specific information needs by posting questions and get direct responses by other users. Harper et al. [8] define Q&A as: *"a web site purposefully designed to allow people to ask and respond to questions on a broad range of topics"*. Bian et al. [9] define Q&A platforms as: *"a form of information retrieval where the users' information need is specified in the form of a natural language question, and the desired result is a self-contained answer (not a list of documents)"*. Examples of traditional Q&A are Quora[4] and Yahoo! Answers[5] where both cover a broad range of topics. Likewise, there are web sites that cover specific areas, such as Stack Overflow[6] whose focus is on computer programming questions. In these web sites, users can rate the content submitted by others. It means that users generate and rate the content themselves.

Although there are Q&A specialized sites for asking questions, people use other media sources. In particular, Social Networks provide a source of information complementary to search engines [4]. In fact, there are people that prefer to pose their questions in social networks rather than search

[4] `http://www.quora.com`
[5] `http://answers.yahoo.com`
[6] `http://www.stackoverflow.com`

engines. They are motivated because of the confidence of their friends and the speed of responses [4].

We acknowledge that there are prior studies on identifying relevant features for cQA retrieval. However, we cannot apply these approaches to other scenarios. For example, a study in QA Yahoo! Answers cannot be fully applied to Twitter because it differ in the types of features that it has. Further, the length of microblog messages is very short in comparison to answers found in cQA platforms. This requires the use of Twitter specific tools and features that differ from prior work. Another example is that users tend to be more specialized in particular topics in cQA platforms, and they ask questions and provide answers for very few topics [10, 11]. Users in Twitter do not tend to be so specific.

For instance, Shtok et al. [12] propose a statistical model to answer new questions asked in the past in Yahoo Answers. The research community of Q&A has expanded the focus on other sources where people generate information needs, and not just in traditional Q&A web sites.

There have been studies that analyze how and what users ask in non-specific QA social networks. For instance, Morris et al. [4] perform 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. [6] conducted a similar study in Twitter and found that the most popular questions are *"rhetorical"* and *"factual Knowledge"*. Zhao and Mei [5] extract certain features from tweets and build a classifier that distinguishes real questions (questions that need answers) from rhetorical questions. In addition, Liu and Jansen [13] create a taxonomy and describe the types of questions that users ask in Twitter.

Another relevant line of research concerns studies on questions and answer pairs in microblogs. Paul et al. [6] observed that roughly 18.7% of Twitter questions receive at least one reply; the first replies come within 5 - 30 minutes and the remainder within the next 10 hours. Liu et al. [14] studied the response rate in the Chinese microblog Sina Weibo and analyzed the characteristics that affect the response rate, such as the number of followers, posting rate, etc. In the same platform, Liu et al. [15] predict potential answerers to questions, based on features such as user characteristics, including popularity. Schantl et al. [16] report that user replies to questions in Twitter are influenced by social relations rather than topics. Schantl et al. [17] report similar results, but observed that users with large network connections, separate them depending on the topic.

Nevertheless, our work focused on conversations that can solve information needs. We define a conversation (or thread) in microblogs as an initial tweet followed by one or more replies (or answers). The initial tweet can be a question or a simple tweet. A conversation is generated when a user replies to a tweet (using the reply option). There are studies that analyze these messages. The streaming nature of this platform makes the data volatile and hence a more difficult problem to address.

To the best of our knowledge there are currently no studies that specifically address identifying relevant conversations in microblogs to satisfy information needs in Twitter. There are related studies that address different aspects of conversations, for example, Boyd et al. [18] study message reposting (or retweets) as if they were conversations, but do not address other types of conversations such as conversation threads. Honey and Herring [3] study conversations in Twitter by defining a conversation as mutual user mentions. Backstrom et al. [19] study conversations in Facebook and predict user participation in other similar conversations.

In our current work we use LTR as a means for learning relevant features for QA using microblogs. In the past, there have been studies that use similar techniques but for different purposes, for example, Duan et al. [20] rank tweets according how informative they are by using content features, such as URLs and tweet length. Also, Molino et al. [21] extract features and use LTR methods to predict *best answers* in Yahoo Answers. In this aspect, we find that features used for Yahoo Answers cannot be directly applied to Twitter, as they are very different platforms. We therefore complement this work by studying which features contribute to finding relevant answers in Twitter and if those features are different from those in other platforms. Surdeanu et al. [22] also study non-factoid questions in Yahoo Answers using several combinations of features and Natural Language Processing (NLP).

Although studies on cQA platforms address factoid and non-factoid questions, the problem of evaluating non-factoid questions for non-specific cQA platforms is difficult in Twitter. In general, for evaluating non-factoid questions in Yahoo! Answers previous work uses answers that have been selected by the community as best-answers, as a ground truth. Therefore, their ranking tasks generally consists of predicting the best answer for a question (from the list of answers available for that question). We do not have this information in Twitter.

To the best of our knowledge, this article presents the first approach to ranking conversation threads using thread features in streaming platforms.

We introduce the initial problem of using microblogs for QA retrieval in [23]. We then study Twitter as QA platform [7]. In particular, we analyze conversation threads on Twitter which provides more information than simple tweets. In addition, we study several sets of features extracted from threads to determine relevance. The present article is an extended version of the latter.

3. Problem Statement

Questions often represent complex information needs which are not easily satisfied with traditional Web information retrieval techniques. For solving this problem, several specialized community QA websites have become very popular on the Web, such as Yahoo! answers and StackExchange. Nevertheless, there is evidence that these sites do not completely solve this need. In a quick inspection, we also found significant QA interaction in microblogs (in particular, in Twitter). Microblogs have the potential of becoming an informal crowd-tailored knowledge base for answering questions online. Although microblogs constitute a historical knowledge base, data are not easy to extract given the characteristics of these kinds of platforms. Moreover, social features of microblogs are not always suitable for QA. Therefore, we address the problem of how to leverage Twitter information for the task of QA retrieval. In particular, we create a ranking model for relevant answers to identify the most important features for effective microblog QA.

Formalization. Let q^* be a question corresponding 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^* which allows us to retrieve a set of conversation threads (i.e., documents) that are candidate answers for q^* [24].

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 conversation thread (hereinafter, a thread) t_i matches a 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, match the initial question q^*. Hence, 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^*.

4. Proposed Solution

Our goal is to evaluate if Twitter data can be used as an additional information source for automatic QA retrieval. To address this, we model our research problem as that of *relevance ranking*, in which the main goal is to rank relevant answers to a query in the top positions. For our current purpose, we define an answer as *relevant* if it contains a correct answer to the original question. Hence, the goal is to rank relevant answers (i.e. correct answers) first, and study which microblog features have a stronger effect on good rankings.

Since microblog messages are of much shorter length than traditional cQA user posts, we propose an aggregated *thread-level document model*. This document representation considers *conversation threads* (as opposed to single tweets) as documents for retrieval. This allows us to preserve the context of the conversations, in which tweets take place, as additional information for answering questions.

In this sense, the scope of our current work is to rank documents (i.e., conversation threads) according to their relevance for a given question. However, we do not address the issue of selecting the single tweet within the conversation thread that contains the exact answer. This task is known in the area of QA as *passage selection*, and constitutes a subtask of the QA problem, which we leave for future work. However, we point out that most conversation threads consist of only one tweet (with no replies), and that those that do have replies usually contain a unique response. This makes it rather easy for the user to identify the answer portions in a thread.

Ideally, our methodology would be applied to the complete Twitter historical database. However, we approximate this by applying our method to the data retrieved using the Twitter Search API as an endpoint, which provides access to a sample of the actual data. This can limit our recall of candidate answers for more uncommon topics.

In particular, we will 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 is both, a quantitative and qualitative analysis, based on a dataset collection retrieved manually. Our intention is to use the factoid task as a proxy to our goal of answering more complex questions (non-factoid) by employing transfer learning, i.e., learning a model for one task (factoid QA) and transferring the model for a different task (non-factoid QA.)

Summarizing, we validate the effectiveness of retrieving relevant answers to questions using past information exchanges on Twitter (**RQ1**). We perform an ablation study to measure the contribution of different features for ranking purposes (**RQ2**), and observe if a thread-level document representation can help to provide answers to non-factoid questions (**RQ3**).

The following three subsections the common steps in the pipeline for both factoid and non-factoid tasks, which are (i) query formulation for thread retrieval, (ii) feature extraction, and (iii) ranking of microblog threads.

4.1. Query Formulation

Query formulation (QF) is the process to create, from the original question q*, a list of keywords that form an IR query [24]. Moreover, query reformulation corresponds to the approach of rephrasing the original query string to make it look like a substring of potential answers [24]. Our approach for retrieving conversational microblog threads with potential answers relies on four strategies for QF, which we describe here. This method increases the recall of documents that may contain an answer to the query q^*.

For each question q^* in the ground truth we retrieve from Twitter its related tweets using our QF in Q^*. If the retrieved tweet is part of a conversation thread we also retrieve its full thread. The complete set of Twitter threads obtained using Q^* corresponds to the set of candidate answers for q^*, $\mathcal{T}$. In addition, we do not apply any filters in the retrieval phase as to discard irrelevant documents beforehand. The QF we use for microblog QA retrieval are:

q_1: Correspond to the original question as it was formulated by the user q^*, without any changes nor filters.

q_2: Correspond to q^* after lowercase and whitespace normalization, removal of non-alphanumerical characters and terms with only one character.

q_3: Correspond to q_2 after the additional removal of stopwords, with the exception of terms in the **6W1H**[7]. For example, the question q^* = *"What is the scientific name of tobacco?"* becomes q_3 = *"what scientific name tobacco"*.

[7] 6W1H correspond to 5WH1 with the addition of the terms *"Which"* (i.e. Who, What, Where, When, Why, Which and How).

q_4: Correspond to q_3 without the **6WH1**. In the previous example, q^* would be transformed to q_4 = *"scientific name tobacco"*.

We apply lowercase normalization and removal of non-alphanumerical characters in q_2 and q_3. Finally, the retrieved tweets are $\{q_1 \cup q_2 \cup q_3 \cup q_4\}$.

We acknowledge that there are other important subtasks in the process of QF for QA, such as creating the best possible query formulations [25] and selecting the passages within a text that contain the answer to a question [24]. However, we consider those problems as beyond the current scope of our research, and them for future work.

4.2. Feature Extraction

We performed a review of features used in prior work on traditional QA, such as [26, 27, 20, 21] and adapted those that could be applied to microblog data. In addition, we proposed some novel features. To do so, we take a well-known question dataset and retrieve similar tweets (based on keywords). In this process, we retrieved approximately 1.000 QA conversation threads in Twitter. This information was useful to qualitatively estimate which features could potentially be influential for determining if the initial question was answered within the conversation thread itself.

In summary, we extract several features of threads based on the state of the art and others that we proposed. Table 1 summarizes the features identified for our general solution, grouped by type. Notice that features are computed at thread level.

- **Word embedding representation (WEMB_*)** : Our study includes a word embedding representation of questions and threads. Several works in NLP [29, 30] have significantly improved their performance using word-vector embeddings such as word2vec [31] rather than the traditional vector space model with TF-IDF weights. In particular, we use Word2Vec [31] with a pre-trained model on 400 million tweets provided by Godin et al. [28]. Each vector was composed of 300-dimensions and since the documents in this model are matrices rather than vectors, we used the max-over-time pooling introduced by Collobert et al. [29] to flatten our document-matrices to document-vectors. We tested with other models such as a pre-trained model

Feature ID	Description
WEMB_Q	Explicit vector representation of the question q^* using word2vec (300 dim.).
WEMB_THR	Explicit vector representation of the thread using word2vec (300 dim.).
D_TFIDF_N (ngram $N = \{1, 2, 3\}$)	Cosine, Manhattan, Euclidean and Jaccard between q^* and the thread t_i.
D_WEMB	Cosine, Manhattan, Euclidean distances between q^* and the thread using word2vec.
SOCIAL	
SOCIAL_N_REPLIES	Number of replies.
SOCIAL_NDIF_REPLIERS	Number of different repliers.
SOCIAL_RATE_FAVORITES	Average of favorites.
SOCIAL_RATE_RETWEETS	Average of retweets.
SOCIAL_MENTIONS	Number of mentions.
SOCIAL_NDIF_MENTIONS	Number of different @mentions.
SOCIAL_NDIF_HASHTAGS	Number of different #hashtags.
USERS	
USERS_NDIF_FOLLOWERS	Number of followers of different users of the thread.
USERS_NDIF_FOLLOWINGS	Number of followings of different users of the thread.
USERS_AVE_AGE	Age average between replies and user date of Twitter.
USERS_RATE_VERIFIED_ACCOUNT	Average of users with verified account.
CONTENT	
CONTENT_NDIF_URLS	Number of different URLS.
CONTENT_N_WORDS	Number of words.
CONTENT_DENSITY	Number of words / number of tweets.
CONTENT_RATE_UPPER	Average of uppercase.
CONTENT_RATE_LOWER	Average of lowercase.
CONTENT_EMOTICONS_POS	Number of positive emoticons.
CONTENT_EMOTICONS_NEG	Number of negative emoticons.
CONTENT_EMOTICONS_NEU	Number of neutral emoticons.
CONTENT_RATE_VOCAB	Rate of well written words.
TIME	
TIME_LIFESPAN	Time difference between the first tweet and the last.
TIME_AVERAGE	Time average between each tweet of the thread.
POS	A set of parts of speech tags.
REPW	Rate of the words that are in the 50% more representative words.

Table 1: Feature sets used for ranking task. In some cases, there are single features and a set in others. For word embeddings (WEMB), we use a pre-trained model of 400 million tweets by [28].

of Google News[8] and word vectors trained on Wikipedia using fastText[9], but after preliminary experiments, Twitter pre-trained model performs better than others. In this case, we use the explicit vector representation of the query (WEMB_Q) and threads (WEMB_THR) using Word2Vec.

- **Distances features (D_TFIDF_N and D_WEMB)**: These features are based on four well-known distance metrics between a thread t_i and a query q^*.
  - Cosine distance:
  $$Cosine(\vec{q^*}, \vec{t_i}) = 1 - \frac{\vec{q^*} \cdot \vec{t_i}}{\|\vec{q^*}\| \, \|\vec{t_i}\|}$$
  - Manhattan distance:
  $$Manhattan(\vec{q^*}, \vec{t_i}) = |\vec{q^*} - \vec{t_i}|$$
  - Euclidean distance:
  $$Euclidean(\vec{q^*}, \vec{t_i}) = \|\vec{q^*} - \vec{t_i}\|$$
  - Jaccard similarity:
  $$Jaccard(q^*, t_i) = \frac{q^* \cap t_i}{q^* \cup t_i}$$

  These features are computed using TFIDF and word2vec representation between the query $\vec{q^*}$ and each thread $\vec{t_i}$. For TFIDF, we denote the computation of these four metrics as **D_TFIDF**_N where N is the ngram used; $N = \{1, 2, 3\}$. For word2vec, we make the same, but excluding Jaccard. We denote it as **D_WEMB**.

- **Social-based features (SOCIAL)**: These features are based on the social interactions observed in a conversation threads (i.e., thread

[8] https://code.google.com/archive/p/word2vec/

[9] https://github.com/facebookresearch/fastText/blob/master/pretrained-vectors.md

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 (USER)**: This feature set considers properties of the users that participate in the same 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. This latter is the difference between date of creation of the Twitter user account and the date when the tweet was posted. We adapt from [26, 27]. The formula is as follows; let $T_w = \{t_1, t_2, \ldots, t_n\}$ a thread with n tweets, $date(t_i)$ is the date when the tweet t_i was published and $UserDate(t_i)$ is the date when the user of the tweet t_i creates an account in Twitter. The user average age (UAA) of the thread is defined by:

$$UAA(T_w) = \frac{1}{n} \sum_{t_1}^{t_n} \Big[date(t_i) - UserDate(t_i) \Big]_{days}$$

  The idea of using the number of followers has been adapted from Duan et al. [20].

- **Content-based features (CONTENT)**: This set of features refers to content properties 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 of uppercase and lowercase letters, the number of positive/negative/neutral emoticons, and the average number of words in English[10].

  These features have been adapted from [20, 21, 26, 27, 9, 32]

- **Parts-of-speech features (POS)**: These features are based on parts of speech tagging. We compute the frequency of each high-confidence POS tag in a conversation thread, using the Twitter-specific tagger *TweetNLP* by Owoputi et al. [33]. For example, the tweet: *"Arya,*

[10]We use the English word corpus of NLTK: `http://www.nltk.org/`.

Sansa and Jon need to reunite. #GameofThrones" has ten tags, where: *"Arya Sansa Jon"* are proper nouns, "." are punctuation, *"and"* is a conjunction, *"to"* is a preposition, *"need reunite"* are verbs, and *"#GameofThrones"* is a hashtag. *TweetNLP* has 25 categories for tagging words. We annotate the tag frequency of each thread with a confidence value greater or equal to 0.7.

The complete list of tagset is in Owoputi et al. [33]. However, some of these tags are related to social: #hashtag, @mention, URLs and emoticons. We decided to remove these tags because we use them in social features. Therefore, we use this set of features with 21 tags.

- **Representative words (REPW)**: This feature corresponds to the fraction of *representative words* that are contained in a thread. *Representative words* are words contained in the top-50% most frequent terms over all threads in the training data (excluding stopwords). We evaluated other variations for this feature (without removing stopwords, using Term Frequency, etc.) and selected the best one.

- **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.

4.3. Ranking of Microblog Threads

We formalized our problem in the previous section as one which allows us to learn a function $f(q^*, \mathcal{T}) \rightarrow \pi$ which ranks conversational microblog threads $\mathcal{T}$. $f(q^*, \mathcal{T})$ gives higher scores (and consequently higher ranks) to threads that correctly answer a question q^*. A popular approach to this problem is Learning-to-rank (LTR). LTR refers to machine learning techniques which learn, from data, a model for ranking items [34]. LTR is widely used in several types of ranking problems in information retrieval (including traditional QA), natural language processing and personalized recommender systems; [22, 35, 36, 37, 20, 38, 39]. We propose to use LTR techniques for learning a microblog QA function, which will also allow us to test which combination of features produces the best overall ranking.

5. Factoid QA Task Evaluation

Jurafsky and Martin [24] defined as *factoid* questions those that require one answer: *"We call the task factoid question answering if the information is a simple fact, and particularly if this fact has to do with a named entity like a person, organization, or location."*. Examples of factoid questions are: *"Who was the first American in space?"*, *"Where did Dylan Thomas die?"*, or *"What is the capital of California?"*. In addition, factoid questions are usually short, much like tweets.

Using different sets of features with LTR methods, we built several models which rank tweets and conversation threads as potential answers to a set of given factoid questions.

A good model ranks relevant tweets or threads at the top of the list, and we defined a tweet or thread as relevant whether it contains the correct answer to a given question.

Therefore, the overall process we carried out is described as follows: given a question q^*, we retrieved similar tweets using four query formulations (QF) and then, for each tweet, we retrieved the complete conversation (if it was part of a thread). We retrieved tweet by tweet to build the thread, since the API does not allow direct retrieval of threads. We then extracted features from threads and label each thread as relevant or not relevant, depending on if the thread contained the correct answer or not. Finally, we trained using features and relevance of threads and generate a model using Learning to Rank (LTR).

To address this problem we used a well known datasets of factoid QA that we describe bellow.

5.1. Ground truth dataset

We built the ground truth QA dataset for our experiments based on several public datasets in the TREC[11] QA challenges[12] and a repository of a factoid-curated questions for benchmarking Question Answering systems[13].

From the TREC we used 4 QA-datasets which contained factoid questions with their respective answers: *TREC-8 (1999)*, *TREC-9 (2000)*, *TREC-2004*

[11] TREC: Text Retrieval Conference

[12] `http://trec.nist.gov/data/qamain.html`

[13] https://github.com/brmson/dataset-factoid-curated

and *TREC-2005*. Three of these datasets, TREC-9, TREC-2004 and TREC-2005, provided factoid questions, their answers, and regular expressions for answers. TREC-8, on the other hand, provided questions and answers, but not the regular expressions for answers, which we reconstructed manually. In addition, the factoid-curated dataset contained factoid questions, their answers and regular expressions for answers. Regular expressions provided for each answers are important because we allow us to perform an automatic evaluation to match correct answers in the retrieved results.

We joined the aforementioned datasets, obtaining a set of $1,634$ factoid questions. Then, we manually eliminated out the following questions (and their respective answers) from this data:

- Time-sensitive questions, i.e., questions with answers that can change over time, for example, *"What is the population of the Bahamas?"*.
- Inaccurate questions. For example, *"what is the size of Argentina?"*.
- Questions not phrased as such. For example, *"define thalassemia"*.
- Questions that required a list of answers. For example, *"What countries has the tsunami struck?"*.
- Questions that referred to other questions. For example, *"What books did she write?"* (it is a reference to other previous question).
- Questions whose length were over 140 characters[14]

After filtering we ended up removing 583 questions, resulting in a final evaluation dataset of $1,051$ questions.

5.2. Tweets/Threads Retrieval

In an ideal scenario, our answers would be obtained from a large historical Twitter dataset, or from the complete data-stream. However these types of data repositories are not available to us at this time. Hence, we approximated them by applying our method to the data retrieved using the Twitter Search API[15] as an endpoint, which provides access to a sample of the actual data.

[14] We are aware that Twitter recently increased the amount of characters to 280. But, we conducted this research before this announcement. Likewise, this change does not affect our study findings.

[15] `https://developer.twitter.com/`

For collecting candidate answers, we used the query formulation procedure described in Section *4.1.* We found candidate answers for 491 (47%) of the questions[16]. For the remaining questions, we cannot found candidate answers (or none of the candidate answers matched the correct answer). Examples of such questions are:

- *"What was the ball game of ancient Mayans called?"*
- *"How many plays were there in Super Bowl XXXIV?"*

For this 491 questions we retrieved a set of candidate answers $\mathcal{T}$ for which at least one of the threads (or tweets) contained the correct answer for q^*. Examples of those questions are:

- *"What city is Purdue University in?"*
- *"When was Queen Victoria born?"*

In addition, to improve and balance the dataset for the learning process, we removed low relevance threads that present all of these three conditions: (1) if it not relevant, (2) if the thread has no replies (just tweets) and, (3) if the cosine distance between the query and the thread is ≤ 0.3.

Table 2 shows a description of the complete final dataset.

Number of TREC Questions	491
Questions with ≥ 50 candidate threads	49
Questions with ≥ 10 candidate threads	146
Questions with < 5 candidate threads	277
Number of threads	33,873
Number of tweets	63,646
% of tweets that are part of a thread	46.7%
Avg. replies per thread	0.9
Number of relevant threads	9,406
Number of not relevant threads	24,467
Number of users involved	38,453

Table 2: Dataset description.

[16]This dataset will be made publicly available in the camera ready version of this article.

We note our initial factoid dataset does not contain current topics, which are much more likely to be discussed in social media (the most recent TREC dataset is from 2005, and the oldest tweet we could retrieve is from 2007). This could explain the low overlap between TREC questions and Twitter, which is characterized for discussing more timely subjects [40]. We elaborate more on this in Section *Discussion*.

5.3. Labeling

We then extract features from threads and label each thread as relevant or not relevant, if the thread contains the correct answer. We were able to automate this labeling process since we have, directly from TREC, the regular expression of the correct answers in our ground truth.

5.4. Feature Extraction

We used the features described in the previous section 4.2. In order to make sure that it makes sense to test all these features in our factoid QA experiment, we inspect whether there are a the correlation between single features and the thread relevance. Table 3 shows some of the best and worst correlation.

Feature	**Specific Feature**	**Corr.**
POS	Proper Noun	0.1565
POS	Numeral	0.1415
D_TFIDF_1	Cosine	0.1332
D_TFIDF_2	Cosine	0.1053
D_TFIDF_3	Cosine	0.1008
...		
POS	Mentions	-0.0817
USERS	USERS_AVE_AGE	-0.0875
SOCIAL	SOCIAL_NDIF_MENTIONS	-0.0894
D_TFIDF_2	Euclidean	-0.1020
D_TFIDF_1	Euclidean	-0.1254

Table 3: Pearson's correlation between each single feature and the thread relevance. High values indicate that the feature is a good predictor of answers. There is no correlation between features and relevance.

Given that there are rather small correlations between single features and relevance, we expect that a hybrid method combining several of these features can perform better than isolated features.

5.5. Learning to Rank Methods

In order to identify the features that produce the best ranking, for the factoid QA task, we evaluate several combinations of sets of features using different Learning to Rank models. In this study, we report the results of four LTR models: MART [41], Ranknet [42], Rankboost [43] and LambdaMart [44]. We give a brief description as follows:

- *Multiple Additive Regression Trees (MART)*, also called Gradient Boosted Regression Trees (GBRT), is a pointwise method, which produces regression trees with the aim of predicting the label of documents. It is a boosting algorithm tree model in which the output of the model is a linear combination of the outputs of a set of regression trees. It can be viewed as a method that performs gradient descent in a function space, using regression trees.

- *Ranknet* is a pairwise method which trains a neural network with gradient descent to obtain a ranking function. By default, it uses a three-layered neural network with a single output node to compare pairs. Ranknet optimizes for (a smooth, convex approximation to) the number of pairwise errors [45].

- *RankBoost* is a pairwise method that solves the preference learning problem. It combines weak rankers in several iterations to get the final ranking function, inspired in AdaBoost [46].

- *LambdaMart* uses gradient boosting trees to optimize ranking metrics as cost functions. It starts initially with a week learner to create an unified strong model [45].

For the sake of brevity, we suggest that the reader looking for more details about these methods checks Liu et al. survey [34].

LTR Parameters. 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[17] for this task. For

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

each combination we computed MRR@10 and $nDCG$@10. It means, the top 10 candidate threads. 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; (a) for Ranknet 100 epochs, 1 hidden layer, 10 hidden nodes per layer and learning rate 5.0×10^{-5}; (b) for MART 1,000 trees, 10 leaves and learning rate of 0.1; (c) for RankBoost 300 rounds and 10 thresholds candidates; and (d) for LambdaMart we use 1,000 trees, 10 leaves, threshold candidates: 256.

Features Combination Heuristic. In addition, to find out the best feature combination, we use the heuristic *Forward Feature Construction* [47]. It is mainly used for dimensionality reduction purposes. The idea is to start with one feature and progressively adding another at a time. We keep the feature (or the set of features) that produces the highest increase in performance.

We adapt this heuristic to our problem. We define it as follows: be f_i a feature set (e.g. parts of speech features), $F = \{f_1, f_2, \dots f_n\}$ the set that contains all of our feature sets, and PBC (initially, $PBC = \emptyset$) 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 again run the same evaluation using the resulting PBC in combination with each remaining feature in F (i.e., $(PBC \cup f_1), (PBC \cup f_2) \dots (PBC \cup f_n)$).
4. We repeat the process from the step (2) until there is no significant improvement in the MRR@10 value.

Through this heuristic it is possible to show the contribution of each feature and then to build the best combination.

5.6. Evaluation Metrics

To reduce the probability of obtaining significant differences among the LTR methods only by chance, we relied on bootstrapping [48]. Rather than having a single train/test split of the dataset, we sample with replacement 30 random collections (we test with cross validation without good performing).

Each collection was then divided into 70% of the questions for training (with their respective tweets/threads) and 30% for testing.

We evaluate different combinations of sets of features in every experiment. We compare the ranking results produced by different LTR models and different feature combinations using two information retrieval metrics:

- **Mean Reciprocal Rank (MRR)** : The MRR is the inverse of the position of the first relevant thread. If there are no relevant threads, the MRR is 0 [49]. Thus, MRR is calculated as:

$$MRR = \frac{1}{r_q}$$

  where r_q is the rank of the first relevant thread. A high MRR implies that the ranking places the most relevant thread near the top of the list.

- **Normalized Discounted Cumulative Gain (nDCG)**: This metric measures the gain of a document discounted by the logarithm of its position. This accumulated gain is high when relevant elements appear at the top of the list and the not relevant elements are placed at the bottom. To calculate $nDCG$, we calculate the DCG of the ranking π. We then calculate the ideal DCG ($iDCG$), that means calculate the DCG considering that all the relevant threads are in the top the list (by [49, 50]).

$$nDCG@k(\pi) = \frac{DCG(\pi)}{iDCG(\pi)}$$

  where,

$$DCG@k(\pi) = \sum_{i=1}^{k} \frac{2^{R_i - 1}}{log_2(1+i)}$$

  R_i is the boolean relevance of thread i and k is the number of thread candidates.

- **Empirical Feature Efficiency**:https://www.overleaf.com/project/5c104d0bbdf21b4b9ddf0
  In addition, we analyze the impact of each set of features in the model. For this purpose, we use the metric *Empirical Feature Efficiency*

(EFE) by [51]. It compares the contribution of each set of feature regarding the best combination.

We adapt this metric to our problem as follows:

$$EFE = \frac{MRR(M^{best}) - MRR(M^{i})}{MRR(M^{best})} \quad (1)$$

Where M^{best} is the MRR@10 of the best combination features and M^{i} is the MRR@10 of each set of features defined in Figure 1.

The metric is expressed in terms of the percentage of the best combination.

5.7. Baselines

We compare our approach to the following methods:

- **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.

- **REPW:** This feature corresponds to the feature REPW. Experimentally, this method behaves as an upper bound of the *Twitter Search* method with the advantage that it can be reproduced.

- **BM25:** The Okapi weighting BM25 is widely used for ranking and searching tasks [50]. We use the BM25 document score in relation to a query.

  The computation of BM25 is as follow: given a query Q with terms $q_1, q_2, ..., q_n$, the BM25(d) score of a document d (in our case, threads) is computed as:

$$\sum_{i=1}^{n} IDF(q_i) \frac{f(q_i) * (k_1 + 1)}{f(q_i) + k_1 * (1 - b + b * |D|/\overline{D_{ave}})}$$

  Where $f(q_i)$ is the number of times that term q_i occurs in document d, D is the number of words of document d, D_{ave} is the average number of words per document. The b and k_1 are free parameters for okapi BM25. In particular, we use $b = 0.75$ and $k_1 = 1.2$, which were reported as optimal for other IR collections [52].

5.8. Results of Factoid QA

Table 4 summarizes the results of more than 700 experiments conducted over several feature combinations and LTR methods. Table 5 shows the p-values of statistical significance testing for differences between LTR models based on single feature sets.

Single Feature Results. 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.

Combination	**MART**	**Ranknet**	**Rank Boost**	**Lambda Mart**
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

Table 4: Factoid task, best combinations of features sets, based on MRR@10, and their percent of improvement over the best single feature set (POS).

	D_TFIDF_1	**D_TFIDF_3**	**POS**	**REPWORDS**	**SOCIAL**	**TIME**
D_TFIDF_3 (0.5123)	0.10					
POS (0.6587)	0.00	0.00				
REPWORDS (0.4810)	0.00	0.00	0.00			
SOCIAL (0.5280)	1.00	0.02	0.00	0.00		
TIME (0.4815)	0.00	0.02	0.00	1.00	0.00	
WEMB_Q (0.4942)	0.00	0.07	0.00	0.61	0.00	0.63

Significant with $p < .05$

Table 5: P-values of multiple t-tests on MRR@10 over single isolated features, with Bonferroni correction. Numbers in parentheses show, for context, the mean MRR@10 of the feature using MART LTR model.

Feature combination results. Table 4 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 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 is MART, especially in the feature set combination results of Table 4. 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 [53], 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 results with the REPW feature (with the exception of RankBoost), as shown in Table 6. This consistent behavior underpins our conclusions in terms of the importance of POS for this task.

		MART		**Ranknet**		**RankBoost**		**LambdaMart**	
	Feature set	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 6: 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 contributions. As we mentioned, we use an additional metric to evaluate the contribution of our set of features. Table 7 shows the contribution of each feature. It seems that social features are very helpful to determine the relevance. In contrast, the contribution of time feature is the lowest. However, we can not eliminate features because each of them contribute to something that others have not incorporated.

Baselines. Table 8 shows the MRR@10 of our method and the baselines. Our best combined LTR model beats all baselines, improving the factoid ranking

Set of feature contribution	EFE
SOCIAL	**8.28%**
D_TFIDF_1	7.38%
POS	3.45%
WEMB_Q	2.18%
D_TFIDF_3	1.13%
REPW	0.58%
TIME	0.47%

Table 7: Contribution of each set of feature regarding the best combination. We use the metric EFE proposed by [51].

results by 74.77% in terms of MRR@10 and by 29.4% on nDCG@10 over Twitter Search. If we compare our results with those on cQA we find similar levels of performance in terms of the MRR metric. Our best model reports a MRR = 0.7795 on the factoid CLEF challenge using Twitter data, while Molino et al. [21] reported a MRR = 0.7954 on their best model on Yahoo! Answers, which can be interpreted as a very similar ranking capacity. For the sake of comparison, previous work by Agichtein [54] reported MRR = 0.6389 and Dalip [36] MRR = 0.7262, respectively.

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**

Table 8: Results of our best combination vs. baselines. We improve over Twitter search up to 74.77% (MRR@10) and 29.4% (nDCG@10).

In addition, Figure 2 shows the ranking quality of baselines and our method from 1 up to 10. Our method performs well at the beginning. The other methods start with a poor performing and then they increase, but never better than our model. We observe that Twitter Search starts with poor performance, but then (at 2) increases the quality and performs better than the other baselines. It means that the ranking method of Twitter Search performs well despite that we do not know precisely how it works. However, our method performs better than all other instances from 1 to 10.

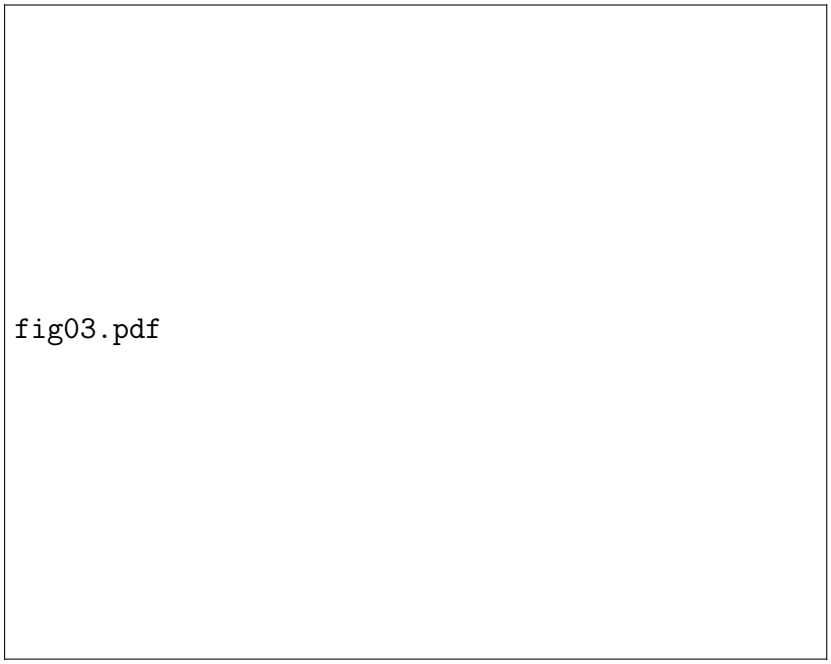

Figure 2: Ranking quality comparison between the baselines and the best feature combination using MART algorithm.

6. Feature Analysis for Factoid QA

We found out special characteristics of questions and threads in or factoid QA dataset; described as follows:

Average Arrival Time of Replies. We found that the average time between replies was low in long threads. In particular, we found 1.525 threads with more than 5 replies. In 85.9% of these threads, the average arrival time for replies was 2 hours. This means that the replies arrive quickly in long threads (long discussions). Figure 3 shows this behavior. In a future work, we could identify these kinds of questions that generate long discussions and estimate the amount of replies.

Query Formulations (QF). We study the relation between the QF and the relevance of threads retrieved. Table 9 shows the amount of relevant/not relevant threads retrieved for each QF. In this inspection, QF_3 and QF_4 retrieve more threads. The combination of QFs, for example QF_1-QF_2, means that a thread was retrieved using these two QF. In these cases, we consider just one

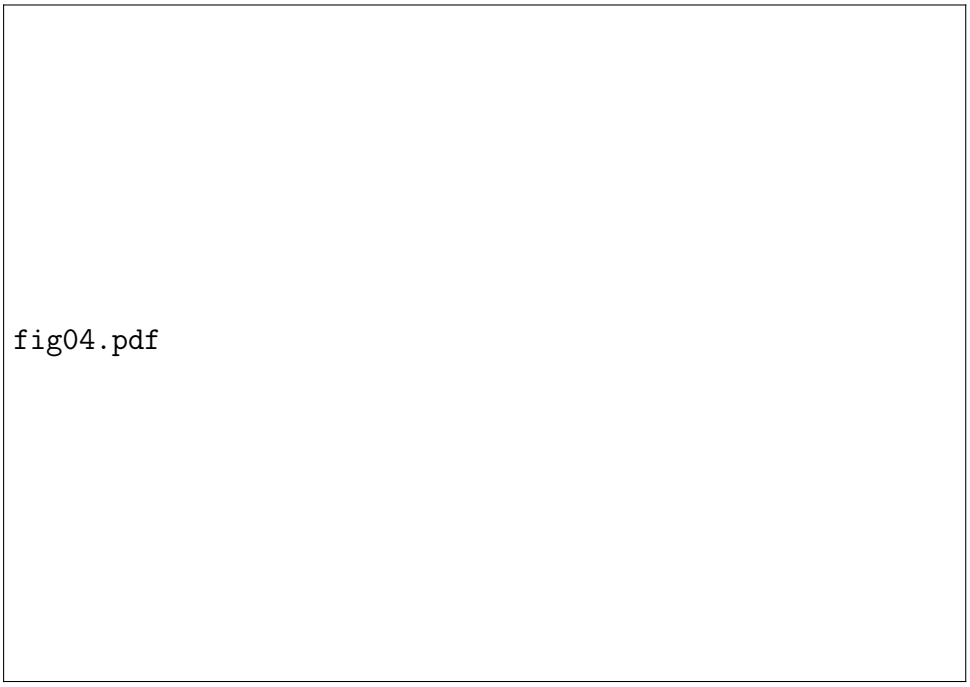

Figure 3: Average arrival time of replies. The 17% of the threads (1,525) had more than 5 replies.

instance. We expected that these QF are the most important because they eliminate noisy words such as stopwords and non-alphanumeric characters. In particular, q_3 retrieved more similar threads related to questions (because it includes 6WH1) and, q_4 retrieved more general threads (without 6WH1). In fact, we could use just these two for the retrieving phase.

nGrams in Distance feature. The best combination (of Table 4) indicates that computing distance with 1 and 3-gram performs well using TFIDF vectors. In contrast, word embeddings have a poor performance in the distance feature. Word embeddings are not appropriate for these distance metrics. In contrast, the TFIDF representation performs better. Likewise, word embeddings perform well when used as an explicit vector of questions or threads.

POS with {2-3}-grams. We addressed additional experiments in order to analyze whether another configuration of POS could improve the ranking process. We study the contribution of POS considering also 2 and 3 grams (bigrams and trigrams, respectively). We determine the top-30 bigrams and

Query Formulations	Relevant	Not Relevant
QF_1	0	6
QF_2	2	15
QF_3	8,421	20,392
QF_4	379	697
QF_1-QF_2	1	10
QF_2-QF_4	8	26
QF_3-QF_4	397	1,805
QF_1-QF_2-QF_4	25	84
QF_2-QF_3-QF_4	84	239
QF_1-QF_2-QF_3-QF_4	89	1,193
	9,406	24,467

Table 9: Amount of relevant and not relevant threads retrieved per QF. For example, using QF_2, we retrieved 2 relevant and 15 not relevant threads; for QF_3-QF_4 (it means that a thread was retrieved using these two QF), we retrieve 397 relevant and 1,805 not relevant threads. NOTE: We omit QFs where we can not retrieve any tweet/thread (for example, QF_1-QF_3).

trigrams of relevant threads. We then run experiments using MART with our POS feature, POS_BI (bigrams) and POS_TRI (trigrams) and the combination between them. Table 10 shows the results of these experiments. The contribution using those three features together is not significant regarding the use of POS isolated (it improves just by 0.91%). For that reason, we consider that computationally, it is not convenient to use more ngrams of POS as a new feature.

Participants and time to get replies. Table 11 shows an evaluation of the characteristics inside the threads. We evaluate three aspects inside the threads; *(1) the user participation*: the number of different users that participate inside the thread, *(2) the time it takes to receive the first reply*: time difference between the initial tweet and the first reply and, *(3) the duration of threads*: time difference between the initial tweet and the last reply.

Honey et al. [3] report a similar number of participant (1) in his study of @mentions in Twitter. In particular, they report a range of $2 - 10$ users per thread and we report $1 - 21$.

Regarding (2), we found out that, in 62% of the cases, the first reply of threads takes less than 30 minutes and 93% takes less than 10 hours. Paul et

	Features	MRR@10 (MART)
(1)	POS (from Table 4)	0.6587
(2)	POS_BI	0.6170
(3)	POS_TRI	0.5806
(4)	POS+POS_BI	**0.6635**
(5)	POS+POS_TRI	0.6630
(6)	POS+POS_BI+POS_TRI	**0.6647**
(7)	POS_BI+POS_TRI	0.6242

Table 10: If we fix POS, the contribution of POS_BI -bigram- is slightly higher than POS_TRI (trigram); (4) and (5), respectively. However, despite that these three features combined present better performance (6), the contribution is not significant (0.91%).

	Range	Median
(1) Number of Participants	1-21	2
(2) Time of the first reply	1 sec - 24 hours	12.8 min
(3) Duration of threads	1 sec - 24 hours	42.1 min

Table 11: Characteristics of conversation threads regarding to user participation and time.

al. [6] reports similar values in their study; 67% of questions receive the first reply within 30 min and the 95% within 10 hours. Moreover, they report a median of 10.3 min to receive the first reply and we report 12.8 min. In short, observing the median, the time to receive the first reply of any thread is around 13 min.

Finally, we report the duration of threads (3). Honey and Herring [3] address a similar study and differ with us. They report a median of 26.33 min in comparison of our 42.1 min. The difference it could be because they consider @mentions as conversations and it does not always represent a conversation thread. In our study, we inspect real conversations as threads taken directly from the Twitter. Likely, they retrieved less data than us (we can conclude that observing the number of participants).

In short, taking the median, conversation threads in Twitter are completed (or satisfied) in around in 42 min. Passed that time, threads are more likely to stop receiving replies. However, the range is wide and could takes till 24 hrs.

Time for the first reply. We made the same previous analysis but now we divide in relevant and not relevant threads. We focus in the first reply of

a thread in order to understand whether relevant threads are more likely to receive soonest replies or not. We observed that the first reply of relevant threads takes longer to arrive compared to not relevant ones. Hence, time seems to be a determinant factor of relevance. In fact, if we observe the previous analysis about the arrival time of the first reply of any thread (Table 11), the result is 12.8 and it is very similar to the not relevant case. It means that relevant threads take longer (about 4 min of difference) to receive the first reply.

6W1H's Contribution. We study the contribution of 6WH1 in thread relevance. Table 12 shows the number of questions that were answered by using our 6WH1 definitions. The majority of relevant threads were asked using *"who"*, *"where"*, *"what"* and *"when"*. Questions with *"how"* and *"which"* do not generate relevant threads, because they are not precise and trigger different kinds of replies which require a semantic interpretation (despite questions are factoid). For instance, *"**How** did James Dean die?"*, the correct answer is *"in a car crash"*, but other valid answers could have been: *"in a car accident"* or *"driving"*. In contrast, question as *"**Who** invented the paper clip?"* has concrete answer: *"Johan Varler"*.

6W1H	# Quest.	# Threads	# Rel	# Not Rel	% Rel
Who	95	7,306	2,465	4,841	33.7%
Where	61	4,410	1,260	3,150	28.6%
What	217	12,265	3,403	8,862	27.7%
When	72	6,742	1,856	4,886	27.5%
How	37	2,487	283	2,204	11.4%
Which	9	619	137	482	22.1%
Why	0	0	0	0	0%

Table 12: Amount of threads retrieved based on 6WH1. Note: We omit 44 threads whose questions did not contain 6W1H.

In short, our method (factoid QA), retrieves more relevant threads when questions are about *"what"*, *"who"*, *"where"* and *"when"*.

7. Non-Factoid QA Task Evaluation

Factoid QA is only one type of task, and recently it has been addressed successfully using deep learning models [55, 56, 57, 58, 59, 60]. Since our

final goal is leveraging all the contextual variables available in microblogging for QA –recency, questions with spatio-temporal context, etc.– beyond factoid QA, we also explore the generalization of the methods and features already analyzed towards non-factoid QA. Unlike factoid QA, Non-factoid QA questions have more than one answer and are usually associated with questions that require opinions, recommendations, experiences, among similar others. Two examples of these kinds of questions are: *"Anyone have any GIF maker software or sites they can recommend?"* and *"Anyone have a remedy for a headache?"*. In particular, we focused on questions related to recommendations which are other types of questions that users request [61, 6, 4]. Morris et al. [4] report that these questions account for a large proportion of questions in Twitter (around 30%).

One important issue with non-factoid QA is the lack of datasets to build a ground truth for training and testing in the way we addressed the factoid QA task. Then, we explore this type of QA through transfer learning, which is a strategy in machine learning *"motivated by the fact that people can apply knowledge learned previously to solve new problems"* [62]. In our case, we collect a small dataset of non-factoid questions and answers and test our existent QA factoid model towards the new task. Our results show that we are able to answer more complex questions using transfer learning, and we provide all the details in this section.

7.1. Tweet/Thread Retrieval

We retrieve recommendation questions from Twitter by the query: *"recommend* ?"* For instance, Figure 1 (in the Introduction) shows non-factoid threads where the initial tweets are these kinds of questions.

To perform a preliminary evaluation of our approach for this task, we sampled 40 diverse non-factoid questions from Twitter. We then follow the same previous pipeline. The only difference is that we use our best model learned in the Factoid QA task. It means that for each query we apply our query formulations (QF), we retrieve threads and we address a feature extraction. Then, we used transfer learning (i.e., we use our best factoid QA LTR model) to rank answers (threads) for the non-factoid task. We generate a ranking of threads using our trained model of the best combination in Factoid QA Task.

The size differences between factoid and non-factoid datasets, shown in Table 13, justify transferring our existing factoid QA model, rather than learning a new one from non-factoid data.

	Non-factoid	Factoid
Number of questions	40	491
Number of tweets	2,666	63,646
Number of threads	386	33,873
% tweets that are part of a thread	87.99%	46,70%
Avg. replies per thread	3.32	0.9

Table 13: Datasets description of non-factoid and factoid QA. Size differences justify the need for transfer learning.

Unlike the TREC dataset of factoid questions, we do not have the ground truth of correct answers. In fact, the correct answer now could be more than one since we are inspecting non-factoid questions. We therefore manually inspected and evaluated the top-10 answers ranked with our approach for each of the 40 questions, labeling them as relevant and not relevant (following the same definition of relevant and not relevant threads of factoid QA).

It was necessary to apply additional filtering in some questions. In the case of factoid questions tasks, we use well-formulated questions from TREC and a curated dataset. But at this time, we retrieve questions directly from Twitter and these are usually miss-written or even present irrelevant information. Therefore, before computing the query formulations of the pipeline, in some questions we apply an additional filtering for removing messages which contain:

- Text as @mentions, #hastags or thanks.
- Irrelevant phrases as *"please, can anyone..."* or *"anyone of my friends..."*.
- Irrelevant adjectives such as *"good"*, *"cheap"*, *"fabulous"*, *"near"*. For example, *"What fun,* ***fabulous*** *restaurant do you recommend in Vegas for dinner, please?"*, *"can anyone recommend me a* ***cheap*** *hotel* ***near the star theatre*** *in singapore?"*.

  We consider using these adjectives to evaluate the strength of questions in a future work.

7.2. Manual Annotation

We annotate several characteristics of threads manually. For each thread, we check whether there were answers inside threads or not. Since we are not

experts in all topics (e.g. hotels or restaurants in unknown cities, anime movies, books recommendation, among others), then we employed valid web sources to ensure answers are correct.

In addition, we annotate other characteristics to evaluate the quality of the results. In particular, we explore ideas from related studies about answer quality addressed in QA. For instance, Jeon et al. [63] studied non-textual features of answers in Naver Q&A (a Korean QA portal). They considered characteristics to predict the quality of answers, such as the answer length, user activity level and number of answers, among others. Shah and Pomerantz [32] addressed a similar study, but evaluating features of answers from Yahoo Answers. Through a human evaluation, they studied 13 different criteria to measure quality of answers. For example, if answers are informative (provide enough information), polite (offending degree), readable and other similar features of [63].

We take the informative metrics of the related studies mentioned above and adapt them to our problem. Although our approach is not directly related to quality of questions or answers in QA with quality, we adapt the metrics to the context of our study. We define two specific heuristics to evaluate the answers[18].

(1) If some tweet in a thread is a **direct answer** to the initial question, it means that the answer is written explicitly inside the thread.

(2) If some tweet in a thread contains an **indirect answer**, i.e., the correct answer is not written explicitly but there is a reference to other sources (such as a URL or a mention to another user).

For instance, Table 14 shows five thread candidates retrieved by our method and which can be answer the initial question: *Can Anyone Recommend a great hotel in Barcelona? 4 daybreak. Thanks.*. We describe each thread as follows:

- Thread 1 has a initial tweet and one reply (R1), and the latter is a **direct answer** (underlined), because it is correct (literal) and valid.
- Thread 2 does not have any replies (we denoted as **thread without replies**), but the initial tweet of the thread contains a **direct answer**.

[18]In this analysis, a **reply** is a tweet which is part of a thread (without counting the initial tweet). Figure 1 shows the parts of a thread.

- Thread 3 have three replies (we denoted as a **thread with replies**), but present one **direct answer** (DA) in the initial tweet.
- Thread 4 is a **thread without replies** and it presents an **indirect answer** (IA), because the answer does not appear explicitly in the tweet (likely inside the URL).
- Thread 5 is a **thread without replies** and also it does not have direct or indirect answer.

Initial Question: *Can Anyone Recommend a great hotel in Barcelona? 4 daybreak. Thanks.*			
THR 1	**Init.tweet:**	-	Hey guys thinking to visit Barcelona. Anyone can recommend great hotel in the centre and places to visit? Planning to visit with my mom.
	R1:	(DA)	Novotel Barcelona City is in the heart of the city on Avenida Diagonal, one of the city's main streets.
THR 2	**Init.tweet:**	(DA)	Btw, if you're heading to Barcelona I definitely recommend the Ohla hotel. Great location, staff, rooms, food `http://t.co/41rB2EKFRE`.
THR 3	**Init.tweet:**	(DA)	Great vacation in Spain. Highly recommend El Palace Hotel in Barcelona (you have to tell cabbies it used to be The Ritz).
	R1:	-	Looking into city break ideas. So far on the list I have Copenhagen, Barcelona and Lisbon. Anywhere else I should be looking?
	R2:	-	Barcelona all the way, I even have a great hotel recommend actually I'll Facebook you...
	R3:	-	Ahhh amazing! Thank you lovely.
THR 4	**Init.tweet:**	(IA)	We totally recommend @axelfriendly hotel in Barcelona and we had a great visit by Maxi `http://t.co/q9ZfP9C7j4`.
THR 5	**Init.tweet:**	-	Still looking for a hotel if anyone has any recommendations?

Table 14: An example of threads retrieved by our method and which could be answer the initial question. Each tweet of threads can be a direct answer (DA), an indirect answer (IA) or a tweet which does not answer the initial question (-).

We carried out a manual evaluation to validate the relevance of the answers, either direct or indirect.

7.3. Results of Non-Factoid QA

Table 15 shows the results of the manual evaluation.

We annotate and evaluate the ranking position of direct and indirect answers. Items 1 to 3 show that there are few threads without replies that incorporate answers. In addition, observing the average replies per threads (Item 7) we notice that it is larger than for factoid evaluation; 0.9 (i.e. a tweet

			Average
Threads per question without replies:			
1)	...	and without direct answers.	4.98
2)	...	with direct answers (or single tweet).	1.15
3)	...	with indirect answers.	0.41
Ranking position of:			
4)	...	the thread with the first direct answer.	2.69
5)	...	replies (inside threads) with the first direct answer.	1.33
6)	...	replies (inside threads) with first indirect answer.	0.87
Replies of a thread:			
7)	...	per thread.	5.00
8)	...	with direct answers.	1.72
9)	...	with indirect answers.	0.56
10)	...	with direct answers and review.	1.10
11)	...	with direct answers, review and URL.	0.11
12)	...	with direct answers with a helpful URL.	0.23
Threads			
13)	...	without replies, without direct answers and before the first thread with direct answer.	1.50

Table 15: Analysis of top-15 threads ranked by our model of non-factoid questions.

with one reply on average). for our factoid dataset and 5 for non-factoid. We can explain this difference considering that non-factoid are more likely to generate more replies (especially in this case of recommendation questions). The position of the first direct answer (Item 4) comes up in the 2nd or 3rd thread, on average.

Regarding the thread replies, the position of the first direct answer occurs generally in the first positions (Items 5 and 6). We also found that around two replies per thread (on average) contain direct answers (Item 8) and around one indirect answers (Item 9). In fact, we found threads without replies (just tweets) but with indirect answers. Moreover, around one reply of a thread (on average) contains a review, experience or additional information related to the direct answer (Item 10).

Surprisingly, we found a low amount of replies contain just URLs for additional or helpful information (item 12). In fact, few replies contained these three elements; direct answer, a review and URL (Item 11).

MRR. We obtained a $MRR@10 = 0.5802$, which is comparable with state-of-the-art results reported recently –MRR=[0.4-0.45] in [53]–, but suboptimal compared to what we obtained in factoid QA tasks.

Improving MRR. By further analyzing the data we found that, on average, for every question we retrieved 1.5 threads without any reply, which were also non relevant to the question made.

We propose a strategy to improve the MRR. Observing the value of Item 13, there are 1.50 threads without replies on average before the first thread with a direct answers. Based on this, we discard from potential answers those threads without replies (or just single tweets). Figure 4 shows an example where four of five threads are removed because they do not have replies. Notice, how the thread number 5 is moved up to first positions improving the MRR.

Figure 4: The thread number 5 has three replies with one indirect answer (R1) and one direct answer (R3). If we consider all threads (without removing), the MRR is 0.2. Removing the first four threads without replies (strikethrough text) the MMR is 1.

We are aware that the proposed solution implies the lost of potentially relevant information. This strategy improved the results to $MRR@10 = 0.6675$, with the small trade-off of one question out of 40 for which we could not find answers. This happens because there are threads without replies but with direct answers. Likewise, the MRR improved by 15% applying the removal process (Table 16). Therefore, this method gives importance to threads that have replies and it seems to be reasonable for recommendation non-factoid QA.

Unlike our factoid QA dataset, non-factoid QA tends to have more replies inside threads (Table 13). We can explain that because users are encouraged to participate giving their recommendations, advising or telling experiences.

Case	MRR	nDCG
1. Our method trained on non-factoid data	0.5802	0.6784
2. Same method after removal of non-relevant threads	**0.6675**	0.7566

Table 16: The MRR@10 and nDCG@10 of our original approach and then after removing threads without replies. For the latter, we eliminated 3 questions where all the threads were removed.

In contrast, factoid answers are concrete and show up earlier in threads (answers are usually in a simple tweet or in top ranked positions of a thread). In fact, the average number of replies for factoid and non-factoid tasks is 0.9 and 3.32, respectively.

8. Discussion

We have studied and demonstrated that microblogs contain valuable information that can be leveraged to obtain answers for information needs paraphrased as questions. The experimental 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 this section, we discuss the results of our research. We focus on the most important aspects of our findings such as the best features, transfer learning and LTR methods.

Feature Importance in Microblog QA. One of the most interesting findings of our evaluation is the high predictive power of POS features. Previous work on community QA conducted on a large dataset of Yahoo! Answers [21] found similar results where proper nouns (such as names of places, people, etc.) and prepositions (such as at, before, on, etc.) are good predictors for relevant answers. It makes sense especially for factoid QA. In addition, we found a good discriminative effect of coordinating conjunctions (and, but, or, so), and we recommend not removing these features as *"stop words"* before conducting POS tagging in factoid QA using microblog data. Figure 5 summarizes and presents the most relevant tags of POS using our factoid QA dataset.

However, we discovered some important differences with the results in Molino et al. [21]. They found punctuation as a discriminative feature between relevant and not relevant answers, whereas it was not helpful in our case. Most Likely, this result is explained by users of traditional QA plat-

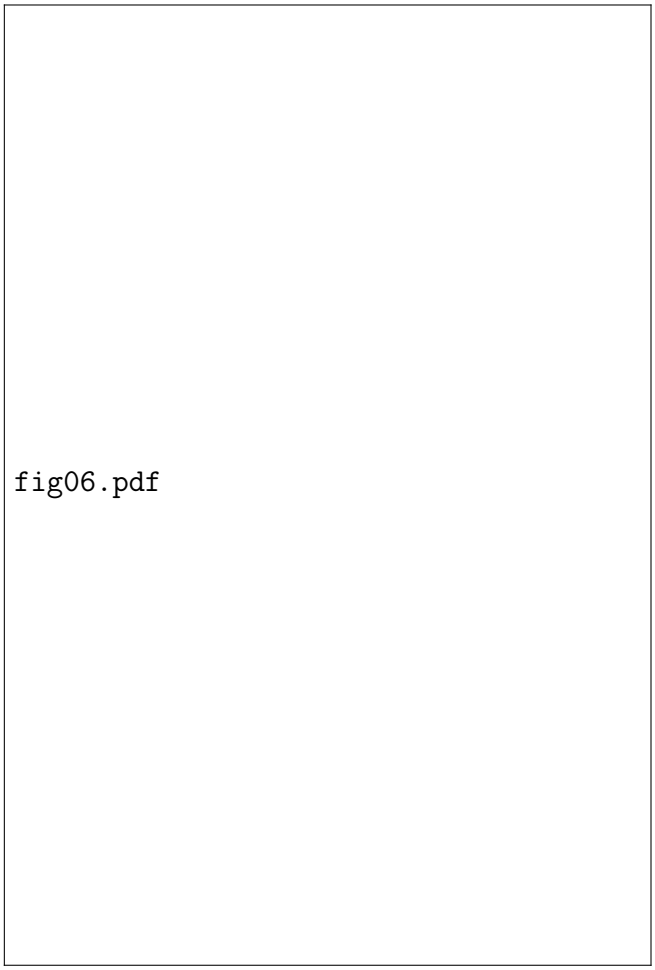

Figure 5: Parts-of-speech (POS) features that discern among relevant and relevant threads. Examples: Common noun: {moon, tree}, Proper noun: {California, Google}, Determiner: {the, a}, Pre-Post position: {of, in}, Coordinating Conj.: {and, or, but} and punctuation: {. :}.

forms, who typically tend to write longer answers compared to microblogs. This phenomena might occur due to space constraints of microblog platforms.

Regarding the contribution of each single feature, Figures 5 and 6 show the most important features for discerning thread relevance.

Surprisingly, social features of Twitter are not good predictors of relevance, with the exception of hashtags. In a inspection of our dataset, we found that threads with hashtags (e.g. #superbowl, #NYC, etc.) are, in

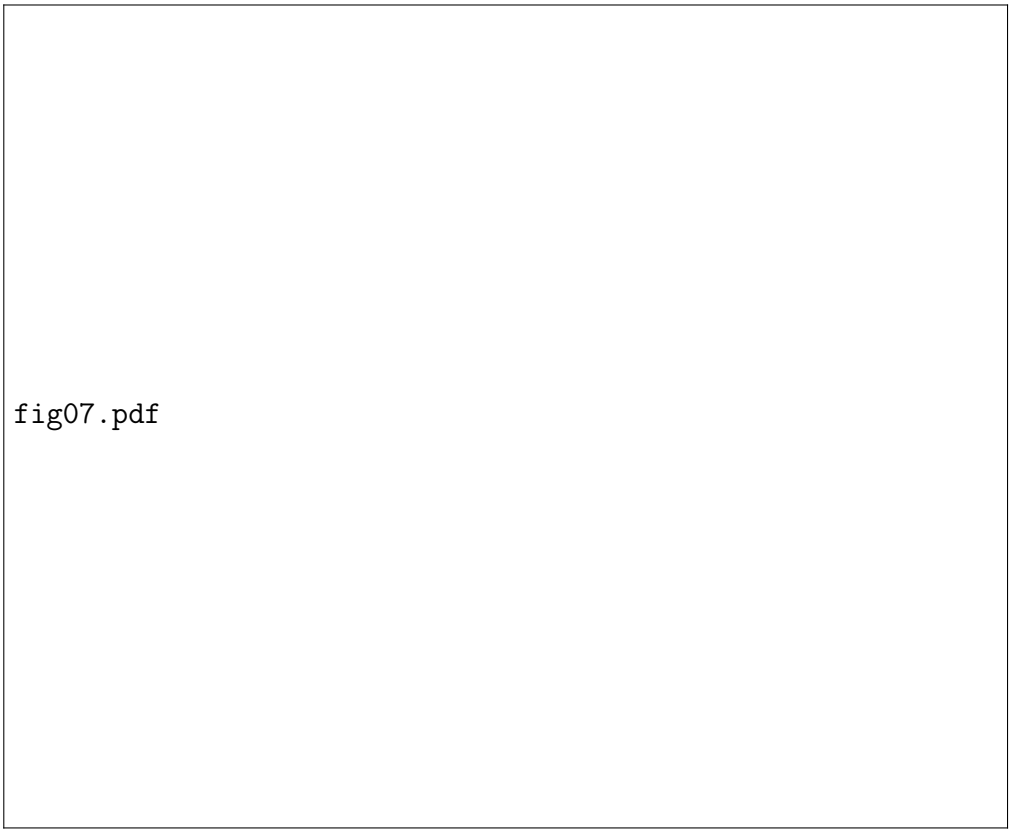

Figure 6: Single features of threads that contribute mostly in the thread relevance.

general, questions that users want to spread beyond his/her network, hence they could tend to receive correct answers. For example, given the question *"How many floors are in the Empire State Building?"*, we found some tweets of threads that contain the following sentences:

```
- How many floors are there in the Empire State Building?
    #EmpireStateBuilding #NYCtrivia #NYC #iloveNY

- How many floors does the Empire State Building have?
    http://t.co/9GNrVWRjXl #uselessfacts, #didyouknow
```

Notice that the meaning of hashtags are about trivia (the first tweet) and to spread though the social network (the second tweet).

In addition, we found that long threads of factoid QA are more likely to be not relevant (number of replies of Figure 6). It happens because factoid

QA questions tend to generate low numbers of replies. When this number is high, conversation threads are related to jokes or other comments between repliers. Similar behavior occurs with @mentions.

On the other hand, we found that relevant threads have a low number of replies but each reply has high number of words (Figure 6). It means that short replies (with concrete answers) are not necessary relevant threads. With these results, we can confirm the findings of Lin et al. [64] who said that users prefer answers in a paragraph rather than the exact phrase.

With respect to time, large differences between replies of a thread indicate that probably the thread is not relevant. We propose to retrieve more of these kinds of features in a future work in order to study the influence of time in some time-sensitive questions, for example, to try to resolve questions quickly in a critical event such as earthquake or terrorist attack.

How to use word embeddings in microblog QA. Another important finding of our study has to do with the best way to use word embeddings for LTR in QA. Molino et al. [21] use this feature as our D_WEMB, i.e., calculating the distance (or similarity) between the query and potential answers based on the skip-gram representation. While they had good results with this feature of *"distributional semantics"*, it was ranked only 30th among other text quality metrics. In our case, we used distances, but also the word embedding representation 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 embedding dimensions as features rather than a single value which aggregates them.

Transfer learning in microblog QA. 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. However, for future work we want to generalize this result and study special features of non-factoid tasks. Moreover, for this experiment we might need to collect a larger non-factoid ground truth dataset.

Overall, the evidence obtained in our current work provides a positive answer for our first and third research questions (**RQ1** and **RQ3**) indicating that Twitter historical data can in fact be used to answer questions, including complex questions related to recommendations. In a future work we propose to incorporate more complex questions such as context-aware (temporal and geo-spatial) as well as personalized. Regarding **RQ2**, our results show that content quality features such as POS play an important role for ranking, even

more than Social and User type features. This is also something that has been observed in other cQA platforms, in relation to other platform specific text-quality features that also provided better indication for finding best answers than other features related to network behavior and user profiles [21].

Limitations. One weakness faced during the factoid QA experiment was that we could only find answers for about 40% of the questions. We analyzed this aspect further, since it is critical for the widespread use of microblog for QA. We note our initial factoid dataset, based on TREC challenges (between 1999 and 2005), does not have topics related to current events, which are much more likely to be discussed in Twitter [40]. The most recent factoid dataset that we used corresponds to the year 2005, however the oldest tweet that we were able to retrieve was from 2007. 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. In general, when examining questions that did not retrieve any candidate answers, we observed that they corresponded to dated topics. Nevertheless, we believe that candidate answer recall could be improved if we used a more complete Twitter dataset, as opposed to a small sample of the data stream as we do now. In this same line of research, a periodic collection of questions asked by users in social media and their corresponding candidate answers, could contribute to creating an up-to-date knowledge base of timely topics.

In addition, the proposed model for factoid questions works with non-factoid, but it needs changes. Non-factoid questions of Twitter present more noise than factoid questions. For future studies we propose to modify (or add) Query Formulations (QF) for this specific task. In addition, we found that threads without replies can be eliminated to improve ranking performance (MMR).

9. Conclusion and Future Work

In this work we proposed the use of microblog data for automatic QA. 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. We show that effective answer retrieval for QA requires different data representation and models than that using traditional cQA platforms.

We studied several sets of features at message level and conversation thread level. We performed a quantitative evaluation on a factoid QA dataset

and a informative evaluation with non-factoid questions. In particular, we found out that parts-of-speech (POS) features are key for determining thread relevance. However, these features require other sets of features such as social and content to perform well. Regarding the LTR ranking framework, MART consistently outperforms the other methods, and the best results are obtained when combining several sets of features.

Regarding non-factoid analysis, out model yield good ranking results (MRR) by performing transfer learning, i.e., by using the same model trained for factoid QA. Moreover, we found out that removing tweets that are not part of a thread, we can improve the MRR.

We note that this is a first look at the use of microblog data for automatic QA. Therefore, our intention was not to find the best possible method for ranking microblog QA answers, but rather to provide evidence of the usefulness of microblogs for effective QA retrieval. Consequently, our baselines were aligned with prior work in cQA research. For future work, it would indeed be interesting to look into baselines based on microblog recommendation, which are not directly adaptable to our current task since they would deviate from our current goals.

Also, in future work we expect to conduct a larger evaluation on non-factoid questions, perform a deeper analysis on the effect of certain attributes, study other features, include other types of questions (not just about recommendations), add new query formulations for non-factoid questions (due the noise of Twitter) and improve our dataset though a crowd sourcing task to generate a real ranking of threads. In addition, we realize that around 50% of our questions were related to places or present an spatial reference. For example, *"Can anyone recommend good accommodation for Paris?"* or *"Anyone wanna recommend any bars in downtown Phoenix?"*. In further analysis we could consider this spatial context to improve the answers.

Another important aspect to consider for future work is incorporating new language models in our framework, by studying the impact of recent neural-based document models such as ELMO [65], BERT [66], TransformerXL [67] and XLNet [68]. These models are progressing quickly and continuously reporting state-of-the-art results in several NLP tasks. Since these neural models not only rank but they also perform feature learning [69], there is a chance that they improve the performance of our current QA framework by creating features as good or better than our current POS tags. Moreover, they can provide further evidence for the suitability of using microblog data for QA tasks.

Acknowledgements

JH, BP and DP have been partially funded by Millennium Institute for Foundational Research on Data; JH was partially funded by the CONICYT Doctoral Program; BP and DP are partially funded by FONDECYT under grants 1191604 and 11150783, respectively.

References

References

[1] M. R. Morris, J. Teevan, K. Panovich, A Comparison of Information Seeking Using Search Engines and Social Networks, in: Proceedings of ICWSM 2010, 2010, pp. 23–26.

[2] D. R. Raban, Self-presentation and the value of information in Q&A websites, Journal of the American Society for Information Science and Technology 60 (12) (2009) 2465–2473. doi:10.1002/asi.21188.

[3] C. Honey, S. C. Herring, Beyond Microblogging: Conversation and Collaboration via Twitter, in: Proceedings of HICSS 2009, ˜IEEE Computer Society, Big Island, HI, USA, 2009, pp. 1–10.

[4] M. R. Morris, J. Teevan, K. Panovich, What do people ask their social networks, and why?, in: Proceedings of CHI 2010, ACM Press, New York, New York, USA, 2010, p. 1739. doi:10.1145/1753326.1753587.

[5] Z. Zhao, Q. Mei, Questions About Questions: an Empirical Analysis of Information Needs on Twitter, in: Proceedings of WWW 2013, University Michigan Ann Arbor, International World Wide Web Conferences Steering Committee, New York, New York, USA, 2013, pp. 1545–1556. doi:10.1145/2488388.2488523.

[6] P. Sharoda, L. Hong, E. H. Chi, Is Twitter a Good Place for Asking Questions? A Characterization Study, in: Proceedings of ICWSM 2011, Barcelona, Spain, 2011, pp. 1–4.

[7] J. Herrera, B. Poblete, D. Parra, Learning to Leverage Microblog Data for QA, in: Proceeding of ECIR 2018, Grenoble, France, 2018, pp. 507–520.

[8] F. M. Harper, D. Raban, S. Rafaeli, J. A. Konstan, Predictors of answer quality in online Q&A sites, in: Proceedings of CHI 2008, Vol. 1, ACM Request Permissions, New York, New York, USA, 2008, pp. 865–874. doi:10.1145/1357054.1357191.

[9] J. Bian, Y. Liu, E. Agichtein, H. Zha, Finding the Right Facts in the Crowd: Factoid Question Answering Over Social Media, in: Proceeding of WWW 2008, 2008, p. 467.

[10] A. Java, X. Song, T. Finin, B. Tseng, Why We Twitter: Understanding Microblogging Usage and Communities, in: Proceedings of KDD (Workshop WebKDD) 2007, ˜ACM Request Permissions, New York, New York, USA, 2007, pp. 56–65. arXiv:1008.1253, doi:10.1145/1348549.1348556.

[11] Z. Gyongyi, G. Koutrika, J. Pedersen, H. Garcia-Molina, Questioning Yahoo! Answers, in: Proceedings of WWW 2008, 2008.

[12] A. Shtok, G. Dror, Y. Maarek, I. Szpektor, Learning From the Past: Answering New Questions with Past Answers, in: Proceedings WWW 2012, ACM, New York, New York, USA, 2012, p. 759. doi:10.1145/2187836.2187939.

[13] Z. Liu, B. J. Jansen, A Taxonomy for Classifying Questions Asked in Social Question and Answering, in: Proceedings of CHI EA 2015, ACM Press, New York, New York, USA, 2015, pp. 1947–1952. doi:10.1145/2702613.2732928.

[14] Z. Liu, B. J. Jansen, Factors Influencing the Response Rate in Social Question and Answering Behavior, in: Proceedings of CSCW 2013, ACM Request Permissions, New York, New York, USA, 2013, p. 1263. doi:10.1145/2441776.2441918.

[15] Z. Liu, B. J. Jansen, Predicting potential responders in social Q&A based on non-QA features, in: Proceedings of CHI 2014, ACM Press, New York, New York, USA, 2014, pp. 2131–2136. doi:10.1145/2559206.2581366.

[16] J. Schantl, R. Kaiser, C. Wagner, M. Strohmaier, The utility of social and topical factors in anticipating repliers in Twitter conversa-

tions, in: Proceedings of WebSci 2013, Graz University of Technology, ACM Press, New York, New York, USA, 2013, pp. 376–385. doi:10.1145/2464464.2464481.

[17] D. Sousa, L. L. L. Sarmento, E. M. Rodrigues, Characterization of the Twitter @replies network: are user ties social or topical?, in: Proceedings of SMUC 2010, University of Porto, ACM, New York, New York, USA, 2010, p. 63. doi:10.1145/1871985.1871996.

[18] D. Boyd, S. Golder, G. Lotan, Tweet, tweet, retweet: Conversational aspects of retweeting on twitter, in: Proceedings of HICSS 2010, IEEE, Washington, DC, USA, 2010, pp. 1–10. doi:10.1109/HICSS.2010.412.

[19] L. Backstrom, J. Kleinberg, L. Lee, C. Danescu-Niculescu-Mizil, Characterizing and Curating Conversation Threads: Expansion, Focus, Volume, Re-entry, in: Proceedings of WSDM 2013, 2013, pp. 13–22. arXiv:arXiv:1304.4602v1, doi:10.1145/2433396.2433401.

[20] Y. Duan, L. Jiang, T. Qin, M. Zhou, H.-Y. Shum, An Empirical Study on Learning to Rank of Tweets, in: Proceedings of COLING 2010, Microsoft Research Asia, Association for Computational Linguistics, Beijing, China, 2010, pp. 295–303.

[21] P. Molino, L. M. Aiello, P. Lops, Social Question Answering: Textual, User, and Network Features for Best Answer Prediction, ACM Transactions on Information Systems 35 (1) (2016) 4–40.

[22] M. Surdeanu, M. Ciaramita, H. Zaragoza, Learning to Rank Answers on Large Online QA Collections, in: Proceedings of ACL 2008, no. June, Columbus, Ohio, USA, 2008, pp. 719–727. arXiv:1412.1632.

[23] J. Herrera, B. Poblete, D. Parra, Retrieving Relevant Conversations for Q&A on Twitter, in: Proceeding of SIGIR 2015 (Workshop of SPS), Vol. 1421, Santiago, Chile, 2015, pp. 21–25.

[24] D. Jurafsky, J. H. Martin, Speech and Language Processing: An Introduction to Natural Language Processing, Computational Linguistics, and Speech Recognition, in: Speech and Language Processing, Vol. 21, Prentice Hall, Pearson Education International, New Jersey, USA, 2009, pp. 0–934. doi:10.1162/089120100750105975.

[25] W. C. Brandão, R. L. T. Santos, N. Ziviani, E. S. de Moura, A. S. da Silva, Learning to expand queries using entities, Journal of the Association for Information Science and Technology 65 (9) (2014) 1870–1883. doi:10.1002/asi.23084.

[26] M. Magnani, D. Montesi, L. Rossi, Conversation retrieval for microblogging sites, Information Retrieval 15 (3-4) (2012) 354–372. doi:10.1007/s10791-012-9189-9.

[27] M. Pennacchiotti, A.-M. Popescu, Democrats, Republicans and Starbucks Afficionados: User Classification in Twitter, in: Proceedings of KDD 2011, ACM Press, New York, New York, USA, 2011, p. 430. doi:10.1145/2020408.2020477.

[28] F. Godin, B. Vandersmissen, W. De Neve, R. Van de Walle, Multimedia Lab @ ACL W-NUT NER Shared Task: Named Entity Recognition for Twitter Microposts using Distributed Word Representations, in: Procceeding of ACL 2015, 2015, pp. 146–153. arXiv:arXiv:1011.1669v3, doi:10.1126/science.1247727.

[29] R. Collobert, J. Weston, L. Bottou, M. Karlen, K. Kavukcuoglu, P. P. Kuksa, Natural Language Processing (Almost) from Scratch, Journal of Machine Learning Research 12 (2011) 2493–2537. arXiv:1103.0398, doi:10.1.1.231.4614.

[30] Y. Kim, Convolutional neural networks for sentence classification, arXiv preprint arXiv:1408.5882 (2014).

[31] T. Mikolov, C. Kai, G. Corrado, J. Dean, Efficient Estimation of Word Representations in Vector Space, NIPS cs.CL (2013) 1–12. arXiv:arXiv:1301.3781v3, doi:10.1162/153244303322533223.

[32] C. Shah, J. Pomerantz, Evaluating and predicting answer quality in community QA, in: Proceeding of SIGIR 2010, Rutgers University, ACM Press, New York, New York, USA, 2010, p. 411. doi:10.1145/1835449.1835518.

[33] O. Owoputi, B. O'Connor, C. Dyer, K. Gimpel, N. Schneider, N. A. Smith, Improved Part-of-Speech Tagging for Online Conversational Text With Word Clusters, in: Proceedings of ACL 2013, no. June, Cornell

University Library, Atlanta, Georgia, USA AD -, 2013, pp. 380–390. arXiv:1011.1669v3, doi:10.1177/001316446002000104.

[34] T.-Y. Liu, Learning to rank for information retrieval, Springer Science & Business Media, Berlin Heidelberg, 2011. arXiv:arXiv:1208.5535v1, doi:10.1145/1835449.1835676.

[35] A. Agarwal, H. Raghavan, K. Subbian, P. Melville, R. D. Lawrence, D. C. Gondek, J. Fan, Learning to Rank for Robust Question Answering, in: Proceedings of CIKM 2012, no. 2, IBM Thomas J. Watson Research Center, ACM, New York, USA, 2012, p. 833.

[36] 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 SIGIR 2013, Dublin, Ireland, 2013, pp. 543–552. doi:http://dl.acm.org/citation.cfm?doid=2484028.2484072.

[37] K. Sun, Y. Cao, X. Song, Y.-I. Song, X. Wang, C.-Y. Lin, Learning to Recommend Questions Based on User Ratings, in: Proceeding of CIKM 2009, ACM, New York, New York, USA, 2009, pp. 751–758. doi:10.1145/1645953.1646049.

[38] H. Zamani, A. Shakery, P. Moradi, Regression and Learning to Rank Aggregation for User Engagement Evaluation, in: Proceedings of RecSys (RecSys Challenge) 2014, ACM Press, New York, New York, USA, 2014, pp. 29–34. doi:10.1145/2668067.2668077.

[39] E. Diaz-Aviles, L. Drumond, L. Schmidt-Thieme, W. Nejdl, Real-time top-n recommendation in social streams, in: Proceedings of RecSys 2012, ACM Press, New York, New York, USA, 2012, p. 59. doi:10.1145/2365952.2365968.

[40] H. Kwak, C. Lee, H. Park, S. Moon, What is Twitter, a social network or a news media?, in: Proceedings of WWW 2010, ACM Press, New York, New York, USA, 2010, p. 591. arXiv:0809.1869v1, doi:10.1145/1772690.1772751.

[41] J. H. Friedman, Greedy Function Approximation: a Gradient Boosting Machine, The Annals of Statistics 29 (5) (2001) 1189–1232. doi:10.3389/fnbot.2013.00021.

[42] C. J. C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, G. Hullender, Learning to Rank using Gradient Descent, in: Proceedings of ML 2005, ACM Press, New York, New York, USA, 2005, pp. 89–96. doi:10.1145/1102351.1102363.

[43] Y. Freund, R. Iyer, R. E. Schapire, Y. Singer, An Efficient Boosting Algorithm for Combining Preferences, Journal of machine Learning Research 4 (2003) 933–969.

[44] Q. Wu, C. J. C. Burges, K. M. Svore, J. Gao, Adapting boosting for information retrieval measures, Journal of Information Retrieval 13 (3) (2010) 254–270. doi:10.1007/s10791-009-9112-1.

[45] C. J. C. Burges, From RankNet to LambdaRank to LambdaMART: An Overview, Tech. rep., . (jun 2010). arXiv:arXiv:1011.1669v3, doi:10.1111/j.1467-8535.2010.01085.x.

[46] Y. Freund, R. E. Schapire, A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting, Journal of Computer and System Sciences 55 (1) (1997) 119–139. doi:10.1006/jcss.1997.1504.

[47] R. Silipo, I. Adae, A. Hart, M. Berthold, Seven Techniques for Dimensionality Reduction, KNIME.com (2014).

[48] B. Efron, R. J. Tibshirani, An Introduction to the Bootstrap, Chapman & Hall/CRC Monographs on Statistics & Applied Probability, Taylor & Francis, Berlin Heidelberg, 1994.

[49] R. Baeza-Yates, B. Ribeiro-Neto, Modern Information Retrieval, Vol. 463, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999. arXiv:9780201398298, doi:10.1080/14735789709366603.

[50] C. D. Manning, P. Raghavan, H. Schütze, Introduction to Information Retrieval, Cambridge University Press, New York, NY, USA, 2008.

[51] L. R. Sánchez, Repurchase Intention for Lodging Recommendation, Phd thesis, . (2017).

[52] M. Surdeanu, M. Ciaramita, H. Zaragoza, Learning to Rank Answers to Non-Factoid Questions from Web Collections., Computational Linguistics 37 (2) (2011) 351–383.

[53] L. Yang, Q. Ai, D. Spina, R.-C. Chen, L. Pang, W. B. Croft, J. Guo, F. Scholer, Beyond Factoid QA: Effective Methods for Non-factoid Answer Sentence Retrieval, Lecture Notes in Computer Science (ECIR 2016). 9626 (2016) 115–128. arXiv:1601.02376, doi:10.1007/978-3-319-30671-1.

[54] E. Agichtein, C. Castillo, D. Donato, A. Gionis, G. G. Mishne, Finding High-quality Content in Social Media, in: Proceedings of WSDM 2008, Emory University, ACM, 2008, pp. 183–193. doi:10.1145/1341531.1341557.

[55] J. Andreas, M. Rohrbach, T. Darrell, D. Klein, Learning to Compose Neural Networks for Question Answering, in: Proceedings of NAACL-HLT 2016, Vol. cs.CL, San Diego, CA, USA., 2016, pp. 1545–1554.

[56] M. Iyyer, J. Boyd-Graber, L. Claudino, R. Socher, H. Daumé III, A Neural Network for Factoid Question Answering over Paragraphs, Proceedings of EMNLP 2014 (2014) 633–644arXiv:1206.5533, doi:10.3115/v1/d14-1070.

[57] L. Yu, K. M. Hermann, P. Blunsom, S. Pulman, Deep Learning for Answer Sentence Selection, NIPS deep learning workshop cs.CL (2014) 9. arXiv:1412.1632.

[58] B. Wang, B. Liu, X. Wang, C. Sun, D. Zhang, Deep Learning Approaches to Semantic Relevance Modeling for Chinese Question-Answer Pairs, ACM Transactions on Asian Language Information Processing 10 (4) (2011) 1–16. doi:10.1145/2025384.2025389.

[59] A. Severyn, A. Moschitti, Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks, in: Proceedings of SIGIR 2015, 2015, pp. 373–382. arXiv:arXiv:1011.1669v3, doi:10.1145/2766462.2767738.

[60] S. Antol, A. Agrawal, J. Lu, M. Mitchell, D. Batra, C. L. Zitnick, D. Parikh, VQA: Visual Question Answering, in: Proceedings of ICCV 2015, IEEE Computer Society, Santiago, Chile, 2015, pp. 2425–2433.

[61] J. Kang, K. Condiff, S. Chang, J. A. Konstan, L. Terveen, F. M. Harper, Understanding How People Use Natural Language to Ask for Recommendations, in: Proceedings of RecSys 2017, RecSys '17, ACM, New York, NY, USA, 2017, pp. 229–237. doi:10.1145/3109859.3109873.

[62] S. J. Pan, Q. Yang, A survey on transfer learning, IEEE Transactions on Knowledge and Data Engineering 22 (2010) 1345—-1359. arXiv:PAI, doi:10.1109/TKDE.2009.191.

[63] J. Jeon, W. B. Croft, J. H. Lee, S. Park, A Framework to Predict the Quality of Answers with Non-Textual Features, in: Proceedings of SIGIR 2006, ACM, New York, New York, USA, 2006, p. 228. doi:10.1145/1148170.1148212.

[64] J. Lin, D. Quan, V. Sinha, K. Bakshi, D. Huynh, B. Katz, D. R. Karger, What Makes a Good Answers? The Role of Context in Question Answering Systems, in: Proceedings of INTERACT 2003, no. September, 2003, p. 1006. doi:10.1145/766117.766119.

[65] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, L. Zettlemoyer, Deep contextualized word representations, in: Proc. of NAACL, 2018.

[66] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, in: Proc. of NACL, 2019, pp. 4171–4186.

[67] Z. Dai, Z. Yang, Y. Yang, W. W. Cohen, J. Carbonell, Q. V. Le, R. Salakhutdinov, Transformer-xl: Attentive language models beyond a fixed-length context, arXiv preprint arXiv:1901.02860 (2019).

[68] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. Salakhutdinov, Q. V. Le, Xlnet: Generalized autoregressive pretraining for language understanding, arXiv preprint arXiv:1906.08237 (2019).

[69] I. Goodfellow, Y. Bengio, A. Courville, Deep Learning, MIT Press, 2016, `http://www.deeplearningbook.org`.