Voice-based Reformulation of Community Answers

Simone Filice
filicesf@amazon.com
Amazon
Seattle, United States of America

Nachshon Cohen
nachshon@amazon.com
Amazon
Haifa, Israel

David Carmel
dacarmel@amazon.com
Amazon
Haifa, Israel

ABSTRACT

Community Question Answering (CQA) websites, such as Stack Exchange[1] or Quora[2], allow users to freely ask questions and obtain answers from other users, i.e., the community. Personal assistants, such as Amazon Alexa or Google Home, can also exploit CQA data to answer a broader range of questions and increase customers' engagement. However, the voice-based interaction poses new challenges to the Question Answering scenario. Even assuming that we are able to retrieve a previously asked question that perfectly matches the user's query, we cannot simply read its answer to the user. A major limitation is the answer length. Reading these answers to the user is cumbersome and boring. Furthermore, many answers contain non-voice-friendly parts, such as images, or URLs.

In this paper, we define the Answer Reformulation task and propose a novel solution to automatically reformulate a community provided answer making it suitable for a voice interaction. Results on a manually annotated dataset extracted from Stack Exchange show that our models improve strong baselines.

CCS CONCEPTS

• **Information systems → Question answering**; **Summarization**.

KEYWORDS

community question answering, deep learning, text summarization

ACM Reference Format:
Simone Filice, Nachshon Cohen, and David Carmel. 2020. Voice-based Reformulation of Community Answers. In *Proceedings of The Web Conference 2020 (WWW '20), April 20–24, 2020, Taipei, Taiwan.* ACM, New York, NY, USA, 7 pages. https://doi.org/10.1145/3366423.3380053

1 INTRODUCTION

In recent years, Community Question Answering became a hot topic in Natural Language Processing (NLP) and Information Retrieval (IR). CQA websites are an extremely valuable source of question-answer pairs and many researchers started leveraging these data into automatic Question Answering (QA) systems.

Given a new user question q, a common QA pipeline based on CQA data consists of (*i*) retrieving previously asked questions $r_1, \ldots, r_n$ highly relevant to q, and (*ii*) distill from their answer threads the information answering q. The second step is usually approached as an answer selection problem [18, 20]: among all the possible answers $a_1, \ldots, a_m$ provided to $r_1, \ldots, r_n$, select the one that best answers q. A possible alternative consists of aggregating $a_1, \ldots, a_m$ in order to provide a more complete answer [23]. In both cases, the selected/aggregated answer can be rather long. For instance, a typical answer in Stack Exchange is longer than hundred words (See Figure 2).

Integrating a CQA system into virtual assistants, such as Amazon Alexa or Google Home, allows to tap into the knowledge of the crowd for answering complex questions. However, this also presents a challenge: how to transform an answer, originally planned to be read on a website, into a voice-friendly answer. One crucial aspect of the reformulation is the need to shorten the community-provided answer. A long answer might be reasonable on a web interface, where a user can easily scroll down, however it might be extremely burdensome to be read aloud fully.

In order to demonstrate the necessity of shortening community provided answers, we measured the distribution of the number of sentences and the number of words in answers taken from subdomains of Stack Exchange, as further described in Section 4. To reduce the amount of unqualified answers in our dataset, we consider only answers accepted by the user who asked the question. Figure 1, presents the distribution of the number of sentences in an answer. There are 48% of the answers longer than six sentences. Figure 2, presents the distribution of the number of words in an answer. The median length of an answer is over a hundred words. If we set the length limit to 55 words (see Section 3), only 24% of the answers can be provided without further summarization.

Another challenge we face is that answers in CQA are often enriched with visual content, such as images or videos, that obviously cannot be directly provided via a voice interface. Furthermore, some textual parts are definitely not voice friendly. For example, it does not make any sense to read an entire URL to the user.

To address these issues, we propose the *Answer Reformulation* (AR) task: given a user question and a long answer retrieved by a CQA system, reformulate the answer to make it more suitable for a voice-based interaction. Our solution consists of a two-step process: (*i*) We first perform a text pre-processing step in order to remove all the answer parts containing non-readable text, including images, videos, URLs, emoticons, etc. (*ii*) We then split the long answer into sentences and invoke a (multi-)sentence selection process to extract a summary consisting of up to 55 words. The latter task uses an extractive-based text summarization approach which selects the most relevant sentences from the answer. We tried different approaches, including pointwise and pairwise learning-to-rank

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

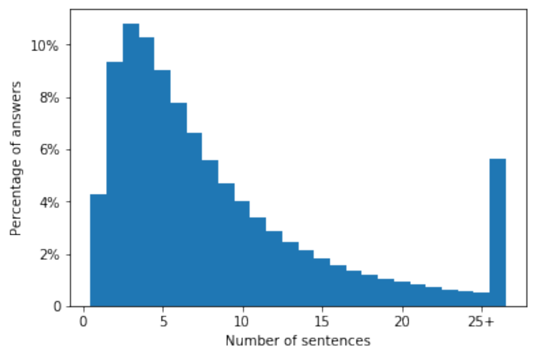


Figure 1: Distribution of answers over number of sentences (median is 5 sentences). The last bar in histogram represents the entire tail, i.e., all answers whose length exceeds the 25 sentences.

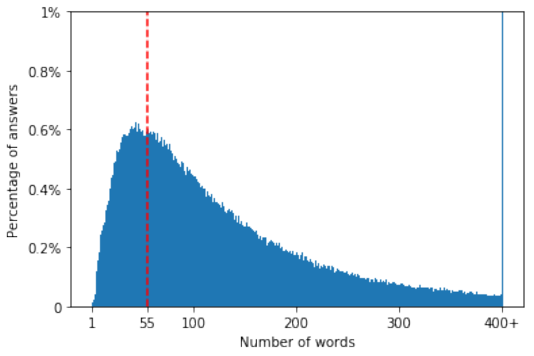


Figure 2: Distribution of answers over number of words (median is 103 words). The red dotted line represents the length limit. The last bar in histogram represents the entire tail, i.e., all answers whose length exceeds 400 words. Histogram is trimmed to 1%: there are actually 6.9% of the answers with over 400 words.

solutions and some sequence tagging methods. In our final model we encapsulate a state-of-the-art NLP solution, i.e., BERT [8], into a novel sequence tagging model that we further enrich with a large set of manually engineered features. To test our models, we ran an annotation task and collected a concise extractive summaries of 2110 answers from Stack Exchange. Experimental evaluation shows that our system performs better than several baselines.

The rest of the paper is organized as follows. Section 2 discusses related work. In Section 3 we present the AR task that we shape as an extractive text summarization problem. We propose several summarization models based on learning-to-rank and sequence-tagging approaches. In Section 4 we describe the data set we use for training and evaluation, and in Section 5 we present our experimental results. Finally, we draw conclusions in Section 6.

2 RELATED WORK

Recently, three consecutive editions of SemEval proposed a challenge on this entire CQA pipeline [18–20]. To automatically answer a new question using the CQA approach, the first step consists of retrieving a set of resolved questions, with their answers, that are relevant to the user question [1, 28, 33]. The retrieval part is typically enhanced with NLP models performing question re-ranking. Many proposed models are based on question-question similarities, including question topic similarity [10] or syntactic similarity [5]. An emerging approach is to apply neural networks. dos Santos et al. [9] used convolutional neural networks (CNNs), while Tan et al. [30] proposed an attention network that applies multiple attention functions to model the matching between a pair of questions.

After finding relevant resolved questions, the answer selection step extracts the answer to the new question from the retrieved question-answer threads. Many approaches were proposed in literature, including deep neural networks [27, 31], kernel methods capturing syntactic-semantic patterns [12] and models based on machine translation [15].

The AR task we propose in this paper is novel, but has strong analogies with query-based summarization and machine reading comprehension. In automatic text summarization [13] the goal is to concisely represent the most important information in documents. The extractive approach [22] consists of selecting (without modifying) the most relevant parts (e.g., sentences, paragraphs, keyphrases, etc.) of one or multiple documents. Conversely, abstractive summarization [21, 26] allows for rephrasing and using words not necessarily presented in the original documents. This second approach is potentially more powerful, as abstraction can condense a text more strongly than extraction; however, it involves natural language generation [14] which is still in its early stages with many open problems such as text quality and naturalness [11]. In our case, we opted for an extractive approach, since text coherency is crucial in voice-based systems. In the query-based summarization [4, 6, 7], the summary must be generated according to a query. Some works are directly applied to CQA and try to aggregate multiple answers into a single, complete summary [29, 32]. In this case, the expected summaries are still rather long and do not fit the constraints imposed in the voice setting.

Finally, machine reading comprehension (MRC) is the task of answering questions about a given paragraph. It is typically approached as a span selection [25] and state-of-the-art systems are based on deep neural networks, e.g., BERT [8]. AR can be approached as a MRC task, if we consider the given answer to reformulate as a paragraph in MRC. The conceptual difference is that in MRC the paragraph should be analyzed to find the answer, while in AR the paragraph (the answer) needs to be summarized to remove redundant content while preserving the important parts.

3 PROBLEM STATEMENT

Given a question with a (possibly long) answer, the AR task consists of transforming the answer into a concise voice friendly version. Although there is no formal definition of what makes a text being voice-friendly, we believe that a voice-friendly answer should adhere to the following properties:

(1) The answer should cover the information need expressed by the question.
(2) The answer should be short enough to be read aloud comfortably. An internal (unpublished) study found that around 55 words is a reasonable length of an answer to be read by a virtual assistant. Therefore, our study considers only summaries with up to 55 words.

(3) The answer should be grammatically correct and coherent.

Based on these requirements, we defined two tasks. The first is a single sentence selection, where the model picks a single sentence to be read as an answer. However, we found that sometimes a single sentence is insufficient in practice since it fails to cover the information need required by the question. Therefore, we also consider multi-sentence selection, which improves information coverage by providing more sentences. We opt for an extractive summarization approach, instead of an abstractive one, to reduce the risk of generating ungrammatical and incoherent answer text.

Our approach consists of (*i*) a text pre-processing step where we clean the answer by removing non voice-friendly elements (such as URLs and pictures) and split the answer into short sentences; (*ii*) an extractive summarization step where we reformulate the answer by selecting the sentence(s) to be read to the user.

3.1 Text Pre-processing

As a first step we remove all the non-readable elements of the answer. This includes images, videos, smileys, and URLs that are replaced with the anchor text[3]. Also, text in parenthesis is removed since it typically contains unnecessary information. Then, the cleaned answer is split into readable units that are sub-selected by our extractive summarization approach described in Section 3.2. Readable units extend the concept of sentence by aggregating sentences that should not be logically split. For instance, two sentences separated by a colon (:) typically need to be read together to express an exhaustive concept. Similarly, it is often better to treat an entire list as an indivisible piece of text, rather than splitting each bullet into a separate sentence. Overall, our text splitting method proceeds as follows:

- The answer is first split into blocks according to the HTML tags (e.g., quote, paragraph, header, sorted and unsorted list, code[4]).
- We split each block using the sentence splitting tool of the CoreNLP library[5], unless the block is already short (i.e., less than 25 words).
- We re-combine blocks that should not be logically separated, e.g., if split by a colon.

Finally, we remove the text formatting HTML tags (e.g., bold, italic) using Jsoup[6]. The resulting blocks are the readable units. For the sake of clarity, in the following we refer to them as sentences.

3.2 Answer Reformulation Models

We shape the AR task as a (multi-)sentence selection problem. To select which sentence(s) should appear in the reformulated answer we propose learning-to-rank and sequence-tagging approaches. We trained our models on data specifically annotated for the AR task.

Learning-to-Rank Approach. We design two learning-to-rank models, a pairwise approach based on the LambdaMART learning algorithm [2], and a pointwise approach based on BERT [8].

From each ⟨*question, answer sentence*⟩ pair we extract several manually engineered features that we use in our ranking models:

(1) Positional features: the position of the sentence inside the long answer, whether it is the first sentence, the last, etc.
(2) Question-sentence lexical similarities : Jaccard, Levenshtein distance, length of longest common substring.
(3) Question-sentence word-embedding similarities: we group the words in the question and in the sentence according to their Part-of-Speech (PoS). In particular, we consider the following groups: nouns only, verbs only, adverbs only, adjectives only, all the remaining PoS. We also consider a group with all words in the sentence. Then, we average the word embeddings from each group and compute the cosine similarities between the resulting question vectors and sentence vectors. We use pre-trained 300 dimensional word embeddings from GloVe [24].
In addition to averaging the word embedding vectors and then computing their similarity, we also compute an *alignment* score: for each word in the question, we find the most similar word in the sentence; both words must have the same PoS group. We then use the average of these maximal similarities as additional features.
(4) Sentence-embedding similarity: the cosine similarity between the question embeddings and the sentence embeddings obtained by using the Universal Sentence Encoder [3].
(5) Lexical features: whether the sentence includes yes/no or summary expressions such as 'in short', 'in conclusion', etc.
(6) HTML and structural features: whether part of the sentence is in bold, is a quote, is a list, was edited (i.e., not included in the first version of the answer), etc.

We use these features to directly represent each sentence in the LambdaMART model. Then the algorithm performs a pairwise reasoning where sentences are analyzed in pairs to establish which one should be ranked first.

In BERT, instead, we apply a pointwise approach: the model receives as input the question followed by a sentence and outputs a relevance score used for ranking. We also extend BERT by injecting our features: we concatenate our features to the first output token of BERT (i.e., the output of the [CLS] token), and run a Multi-Layer Perceptron (MLP) on the resulting vector.

Sequence-Tagging Approach. One limit of the learning-to-rank models described above is that they provide a relevance score to a given sentence while ignoring the context in which the sentence appears (i.e., the rest of the answer). In general, also for a human, it is very hard to assess whether a sentence is the best one without reading the other sentences of the answer. To overcome this issue, we propose a sequence-tagging solution where the entire sequence of sentences is provided as input to a model that tags each sentence as relevant or not to the question.

The entire architecture is depicted in Figure 3. Each sentence in the answer is paired with the question and provided as input to BERT. For each pair, the first output token of BERT is considered. This can be interpreted as a question-aware sentence embedding. Each sentence embedding is concatenated with the manually engineered features described above. Then the sequence of augmented sentence embeddings is passed to a bidirectional LSTM to get a sequence of contextualized sentence embeddings. Finally, inspired by Named Entity Recognition models [16], we run a Conditional

[3]When URLs do not have any anchor text, they are replaced with the phrase "a website".
[4]Code environment is often used to highlight a text.
[5]https://stanfordnlp.github.io/CoreNLP/
[6]https://jsoup.org/

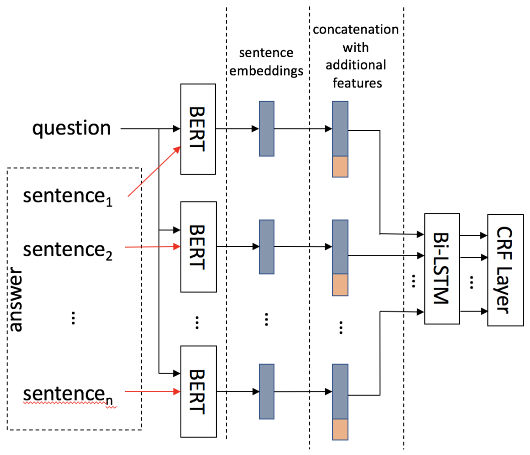


Figure 3: Our sequence-tagging model for answer reformulation.

Random Field (CRF) layer to tag each sentence as being relevant of not. The CRF layer should allow the model to leverage dependencies between successive labels.

4 DATASET CREATION

We collected a set of question and answer pairs from 10 subdomains of Stack Exchange. We explored a Stack Exchange dump containing more than 200k questions. The average length of an accepted answer is presented in Figures 1 and 2.

A training example for the AR task is a triple ⟨question, original answer, short answer⟩. A preliminary requirement is that the original answer completely answers the question. After-all, reformulation cannot transpose a bad answer into a good one. To ensure the quality of the original answers in our dataset, we considered only answers that were marked as accepted by the user asking the question. In addition, we also required that the answer received some up-votes from users in the CQA website (our current system uses 7 up-votes as a threshold[7]). We considered only answers with at least 3 sentences and more than 55 words.

Since we operate in a voice setting we also expect the training questions to represent typical user questions submitted to digital assistants. However, questions in web fora tend to be very long and verbose, and many questions can only be fully understood when considering the question's title and body together. To automatically get questions that better represent voice-based questions, we first randomly selected 3543 question titles ending with a question mark. Then, we verified the consistency and clarity of the selected titles by running a crowd-sourcing annotation task using figure-eight[8]. In particular, we asked to annotate whether the title is self-explanatory, i.e., it can be read as a standalone question. We obtained 2110 ⟨question, answer⟩ pairs for which the question title is self-explanatory. On average, full answers contain 8.45 sentences, resulting in a total of 17,819 sentences in our data set.

[7]We chose the highest possible value that allowed us to collect a number of QA pairs compliant with our budget.
[8]https://figure-eight.com

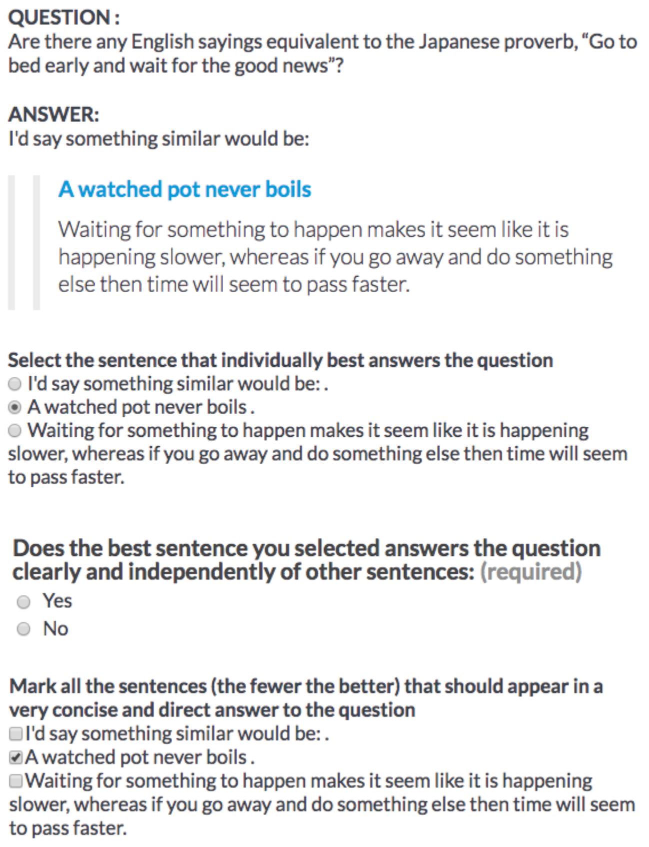
Finally, we run another annotation task to get the short versions of the selected answers. Most of the literature in text summarization requires the annotators to write a reference summary from scratch; however, since we want to apply an extractive summarization approach, we decided to shape the annotation task as a sentence selection problem. Each annotator was presented with the short question (i.e., the question title) and its answer. We then asked the annotators to solve three tasks:

(1) Pick a single sentence that best answers the question.
(2) Mark whether the single sentence you picked fully answers the question.
(3) Pick all sentences that should appear in a short and concise answer.

A picture from the annotation task page is presented in Figure 4. Every ⟨question, answer⟩ pair was annotated by at least five annotators. The single sentence selection (task 1) aims to create the shortest possible version of the answer, and task 2 is used to verify whether such individual sentence is enough for answering the question. Our annotations show that 89.8% of the times a single sentence can fully answer the question. Finally, the multiple-sentence selection task aims to create a more valid and complete summary. We did not put any constraint on the number of selected sentences, but encouraged the annotators to select as few sentences as possible.

Generating ground truth. According to our annotation tasks, we created two types of labels. First, we considered the single-sentence selection task. We filtered out all questions for which a single sentence is insufficient. Then, we picked the most voted sentence by the annotators as the ground truth selected answer. Annotator agreement (w.r.t. to the most voted sentence) for this task was 79.7% and the ground truth was picked by at least two annotators, so this label can be considered reliable.

The second ground truth label is generated from the multi-sentence annotation task. In this task, annotators are free to choose any combination of sentences from the answer, making the range of options very large. Picking the most voted selection as reference summary is generally ineffective: it is likely that each combination of sentences is picked by exactly one annotator. Selecting the most voted sentences, or all the sentences receiving at least a certain number of votes are other possibilities that we decided to exclude. The reason is that such choices can generate a new selection that none of the annotators exactly picked, and it would expose to the risk of selecting a sequence of incoherent and possibly redundant sentences. We solved this problem by picking the selection of a single annotator that is the most representative of the annotators' opinion. More specifically, we created a random sentence selection model, where the probability of selecting a sentence is equal to the fraction of annotators picking it. Thus, if a sentence s is picked by two annotators out of five, the probability of picking s is set to $p(s) = 0.4$ and the probability of not picking s is set to $1 - p(s) = 0.6$. Using this random model, we assign a probability score to each of the annotated summaries. Eventually, we pick the summary with highest likelihood as ground truth. More formally, let N be the set of sentences in the answer, $\mathcal{S}$ be the set of different summaries selected by the annotators, and I_S and E_S the sets of sentences included and excluded in a summary $S \in \mathcal{S}$, respectively. Then the selected ground-truth summary S^* is:


QUESTION :
Are there any English sayings equivalent to the Japanese proverb, "Go to bed early and wait for the good news"?

ANSWER:
I'd say something similar would be:

A watched pot never boils

Waiting for something to happen makes it seem like it is happening slower, whereas if you go away and do something else then time will seem to pass faster.

Select the sentence that individually best answers the question
- I'd say something similar would be: .
- [✓ A watched pot never boils .]
- Waiting for something to happen makes it seem like it is happening slower, whereas if you go away and do something else then time will seem to pass faster.

Does the best sentence you selected answers the question clearly and independently of other sentences: (required)
- Yes
- No

Mark all the sentences (the fewer the better) that should appear in a very concise and direct answer to the question
- I'd say something similar would be: .
- [✓ A watched pot never boils .]
- Waiting for something to happen makes it seem like it is happening slower, whereas if you go away and do something else then time will seem to pass faster.

Figure 4: The annotation task presents a question and an answer and asks three questions: (*i*) select a single sentence, (*ii*) does the single sentence fully answers the question, and (*iii*) mark all sentences that should appear in a summary.

$$S^* = \underset{S \in \mathcal{S}}{\operatorname{argmax}} \left(\left(\prod_{s_i \in I_S} p(s_i) \right) \left(\prod_{s_e \in E_S} (1 - p(s_e)) \right) \right) \tag{1}$$

To assert the quality of our ground-truth selection, we randomly sampled 100 selected summaries (according to Eq. 1) and manually annotated their quality. We found that in 86% of the cases the quality of the summary was excellent, in 10% of the cases the summary was relevant but imperfect, and in other 4% of the cases it was wrong. Therefore, we consider our technique as a reasonable way to get ground truth labels for our dataset. Table 1 reports the statistics of the created dataset.

5 EXPERIMENTAL EVALUATION

In this section, we evaluate the quality of the proposed answer reformulation models on the dataset extracted from Stack Exchange.

5.1 Experimental Setting

We divided the dataset described in Section 4 using 70% of the questions as training set, 10% as development set to tune our models, and 20% as test set. According to the two different labeling schemas described in Section 4, we defined two sentence selection tasks. In the single sentence selection task, the model has to pick the single sentence that best summarizes the answer. For this task, given the sentence ranking provided by our models, we consider *Precision@1* (fraction of test-cases where the top ranked sentence was marked as best sentence by the annotators) and *NDCG@*{3, 5} (normalized discounted gain for top 3, 5 ranked sentences).

Table 1: The annotated Stack Exchange data. Each example is a triple (Question, Answer, Reference Summary). Q is the number of questions, S is the number of sentences, and W is the number of words.

		Entire Answer		Reference Summary	
Domain	Q	Avg. S	Avg. W	Avg. S	Avg. W
Bricks	24	7.46	123.83	1.21	22.46
Do it yourself	96	8.09	145.07	1.38	25.27
Earth science	38	11.32	204.32	1.37	23.21
Economics	6	13.00	273.50	1.00	30.33
English	901	7.71	141.53	1.50	28.80
Gardening	33	9.45	169.55	1.79	26.15
History	239	9.62	199.59	1.38	30.17
Literature	11	8.73	172.18	1.00	23.64
Movies	535	8.73	170.19	1.35	28.43
Space	227	8.92	181.90	1.48	33.03
Total	2110	8.45	161.78	1.44	28.92

In multi-sentence selection task, we pick multiple sentences, up to 55 words, according to the ranking order. Ground truth for this task is derived from the third question in our annotation task (see Figure 4). Note that for each sentence, we have a ground truth label (i.e., whether this sentence was picked by the annotator) and system prediction (i.e., whether this sentence belongs to the system's selected summary). Thus, we consider sentence-level metrics, i.e., *Precision*, *Recall*, *F1*, and *Accuracy*. We also measure *Success* (fraction of test cases for which the system picked exactly the same sentences included in the reference summary). Furthermore, for both tasks, we adopt common Automatic Text Summarization metrics, i.e., *ROUGE-1*, *ROUGE-2* and *ROUGE-L*.

5.2 Evaluated models

In addition to our models, we consider four popular baselines.

- Lead: we pick the first sentence(s) of the answer, up to 55 words.
- Jaccard: we rank sentences according to their Jaccard similarity to the query.
- TextRank: we rank sentences according to TextRank [17], an unsupervised learning approach based on the PageRank algorithm[9].
- BERT: we use the uncased pre-trained base version of BERT[10] [8]. The input of BERT is a question followed by a sentence and we use BERT as a binary classifier to provide a relevance score to the sentence.

Regarding our models we tested the following solutions:

- LM: the LambdaMART pairwise learning-to-rank model operating on our features.
- MLP: a Multi-Layer Perceptron (MLP) operating on our features.

[9]We used the Sumy implementation: https://github.com/miso-belica/sumy/tree/master/sumy.
[10]In our preliminary experiments we observed that using the cased or large versions was not beneficial.

- BERT$^+$: the extension of BERT in which the output is concatenated with our features and provided as input to a MLP that performs the final classification.
- SeqBERT: this is the sequence-tagging model depicted in Figure 3 where our features are not employed.
- SeqFeats: this is the sequence-tagging model where BERT is not employed: our feature vectors are directly passed to the Bi-LSTM in Figure 3.
- SeqBERT$^+$: this is the complete sequence-tagging model in Figure 3 where both BERT and our features are employed.

The relevance scores provided by these models are used to rank the sentences and generate the concise answer by selecting the top ranked sentences, up to 55 words.

For all the supervised models we use an early stopping strategy on the dev set; for LambdaMART, the number of trees was set to 100 and the number of leaves to 10. The number of LSTM units in the sequence-tagging models is set to 200. In all the models using BERT (including the sequence-tagging ones) we used two different learning rates: one for the BERT layers which must be relatively very small to prevent the model to forget what it learned during the BERT pre-training; for the other layers (i.e., the output layers) we used a more aggressive learning rate, since those layers are randomly initialized and must be trained from scratch. We selected the learning rates and the number of layers in the MLP models by using a grid search on the dev set. For the best sentence selection task, we pick the configuration providing the best NDCG@3[11], while for the multi-sentence selection task we optimize per sentence accuracy.

Table 2: Single sentence selection. The bold suggests a higher score compared to the best baseline (i.e., Lead baseline) and "*" indicates that the improvement is statistically significant (p<0.05).

Model	Precision@1	NDCG@3	NDCG@5	ROUGE-1	ROUGE-2	ROUGE-L
Lead	0.6921	0.8027	0.8275	0.734	0.684	0.729
Jaccard	0.3632	0.5638	0.6386	0.462	0.38	0.447
TextRank	0.2447	0.3924	0.4828	0.34	0.26	0.322
BERT	0.5092	0.6826	0.7319	0.606	0.539	0.596
LM	0.6816	0.8017	**0.8352**	0.725	0.674	0.719
MLP	0.6921	**0.8051**	**0.8334**	0.734	0.684	0.729
BERT$^+$	0.6895	**0.8238***	**0.8480***	**0.735**	0.683	0.728
SeqBERT	**0.7000**	**0.8323***	**0.8523***	**0.743**	**0.692**	**0.736**
SeqFeats	**0.6947**	**0.8121**	**0.8394***	**0.737**	**0.687**	**0.731**
SeqBERT$^+$	**0.7079**	**0.8336***	**0.8591***	**0.751**	**0.7**	**0.745**

5.3 Results

Single-Sentence Selection. Table 2 reports results for the single sentence selection task. The Lead baseline provides excellent results: in over 69% of the cases, the first sentence is a good summary of the answer. Therefore, it is difficult to improve upon this baseline. Other baselines, based on Jaccard similarity, TextRank, or BERT provide significantly lower scores. One possible reason for the low result of BERT is that when it classifies a sentence, it does not have access to the context in which it appears (i.e., the rest of the answer). The systems based on our features provide more competitive results. In particular, BERT$^+$, by jointly leveraging our features and the deep semantic information extracted by BERT, significantly outperforms the baselines in both NDCG@3 and NDCG@5. Finally the sequence-tagging approach can further improve the results: when the model can access to the entire answer, a pure text-based model such as SeqBERT is already competitive. When our features are also enabled in SeqBERT$^+$ the best results are reached, improving the baselines in all metrics.

[11]According to our experiments NDCG@3 is much more robust for training than P@1. We also noticed that training the models according to NDCG@3 or NDCG@5 does not make much difference.

Table 3: Multi-sentence selection results. The bold suggests a higher score compared to the Lead baseline and "*" indicates that the improvement is statistically significant (p<0.05).

Model	Success	Precision	Recall	F1	Acc.	ROUGE-1	ROUGE-2	ROUGE-L
Lead	0.709	0.432	0.779	0.515	0.713	0.811	0.766	0.805
Jaccard	0.493	0.347	0.583	0.398	0.684	0.67	0.586	0.657
TextRank	0.242	0.259	0.313	0.258	0.672	0.478	0.348	0.454
BERT	0.604	0.423	0.7	0.488	**0.729**	0.752	0.688	0.742
LM	0.694	**0.461***	0.775	**0.536***	**0.744***	**0.812**	0.766	0.805
MLP	0.701	**0.439**	0.777	**0.519**	**0.721***	0.81	0.766	0.804
BERT$^+$	0.694	**0.468***	**0.792**	**0.544***	**0.742***	**0.824**	**0.781**	**0.819**
SeqBERT	**0.713**	**0.468***	0.772	**0.538***	**0.744***	0.808	0.762	0.803
SeqFeats	**0.711**	**0.465***	**0.79**	**0.54***	**0.737***	**0.826**	**0.784**	**0.82**
SeqBERT$^+$	**0.72**	**0.468***	**0.791**	**0.543***	**0.742***	**0.824**	**0.78**	**0.818**
Upper-Bound	0.898	0.578	0.954	0.661	0.8	0.958	0.941	0.958

Multi-Sentence Selection. In Table 3 we present results for the multi-sentence selection task. The summaries extracted by our models are selected by picking the most scored sentences until reaching the 55-words limit. Since we did not impose any limit on the number of words or sentences that an annotator could include in her summary, the reference summaries can be shorter or longer than 55 words. Therefore, even if we select sentences up to 55 words using a perfect ranker, we can create summaries which are longer than the reference ones: all the extra sentences are evaluated as false positive and thus penalize the model precision. Similarly, when the reference summary is longer than 55 words, any model-based summary will be penalized in recall. The upper-bound in Table 3 reports the performance achievable by a perfect ranker. Again, we observe that the Lead baseline provides strong results while Jaccard and TextRank perform rather poorly. In this task BERT achieves relatively better results than in the previous task. This may be justified by the fact that understanding whether a sentence is relevant to answer a question is generally easier than recognizing the best sentence in the answer. Again, the sequence-tagging approach and the usage of our features lead to significantly better results. In this task, SeqBERT and SeqFeats lead to very similar results. The large improvement provided by Deep Learning models that we usually observe in many NLP tasks is here not confirmed. One possible reason is that the dataset size is rather small considering the task complexity. We will better study this aspect in future work.

6 CONCLUSION

In this work, we presented the *answer reformulation* task: how to transfer a traditional community question-answering system into the world of voice assistants. We created an annotated dataset[12] and proposed a novel sequence-tagging model that outperforms strong baselines by jointly leveraging BERT and a rich set of manually engineered features.

[12]anonymized_url

REFERENCES

[1] Eugene Agichtein, David Carmel, Dan Pelleg, Yuval Pinter, and Donna Harman. 2016. Overview of the TREC 2016 LiveQA Track. (2016).

[2] Christopher J. C. Burges. 2010. *From RankNet to LambdaRank to LambdaMART: An Overview*. Technical Report. Microsoft Research. http://research.microsoft.com/en-us/um/people/cburges/tech_reports/MSR-TR-2010-82.pdf

[3] Daniel Cer, Yinfei Yang, Sheng-yi Kong, Nan Hua, Nicole Limtiaco, Rhomni St. John, Noah Constant, Mario Guajardo-Cespedes, Steve Yuan, Chris Tar, Yun-Hsuan Sung, Brian Strope, and Ray Kurzweil. 2018. Universal Sentence Encoder. *CoRR* abs/1803.11175 (2018). arXiv:1803.11175 http://arxiv.org/abs/1803.11175

[4] Yllias Chali and Sadid a. Hasan. 2012. Query-focused Multi-document Summarization: Automatic Data Annotations and Supervised Learning Approaches. *Natural Language Engineering* 18, 1 (Jan. 2012), 109–145. https://doi.org/10.1017/S1351324911000167

[5] Giovanni Da San Martino, Alberto Barrón Cedeño, Salvatore Romeo, Antonio Uva, and Alessandro Moschitti. 2016. Learning to Re-Rank Questions in Community Question Answering Using Advanced Features. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management (CIKM '16)*. ACM, New York, NY, USA, 1997–2000. https://doi.org/10.1145/2983323.2983893

[6] Hoa Trang Dang. 2006. Overview of DUC 2006. In *In Proceedings of HLT-NAACL 2006*.

[7] Hal Daumé-III. 2009. Bayesian Query-Focused Summarization. *CoRR* abs/0907.1814 (2009). arXiv:0907.1814 http://arxiv.org/abs/0907.1814

[8] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding. *CoRR* abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805

[9] Cicero dos Santos, Luciano Barbosa, Dasha Bogdanova, and Bianca Zadrozny. 2015. Learning Hybrid Representations to Retrieve Semantically Equivalent Questions. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Association for Computational Linguistics, 694–699. https://doi.org/10.3115/v1/P15-2114

[10] Huizhong Duan, Yunbo Cao, Chin-Yew Lin, and Yong Yu. 2008. Searching Questions by Identifying Question Topic and Question Focus. In *ACL 2008, Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics, June 15-20, 2008, Columbus, Ohio, USA*. 156–164. http://www.aclweb.org/anthology/P08-1019

[11] Ondřej Dušek, Jekaterina Novikova, and Verena Rieser. 2018. Findings of the E2E NLG Challenge. In *Proceedings of the 11th International Conference on Natural Language Generation*. Tilburg, The Netherlands. https://arxiv.org/abs/1810.01170 arXiv:1810.01170.

[12] Simone Filice and Alessandro Moschitti. 2018. Learning pairwise patterns in Community Question Answering. *Intelligenza Artificiale* 12, 2 (2018), 49–65. https://doi.org/10.3233/IA-170034

[13] Mahak Gambhir and Vishal Gupta. 2017. Recent Automatic Text Summarization Techniques: A Survey. *Artif. Intell. Rev.* 47, 1 (Jan. 2017), 1–66. https://doi.org/10.1007/s10462-016-9475-9

[14] Albert Gatt and Emiel Krahmer. 2018. Survey of the State of the Art in Natural Language Generation: Core tasks, applications and evaluation. *J. Artif. Intell. Res.* 61 (2018), 65–170. https://doi.org/10.1613/jair.5477

[15] Francisco Guzmán, Shafiq R. Joty, Lluís Màrquez, and Preslav Nakov. 2017. Machine Translation Evaluation with Neural Networks. *CoRR* abs/1710.02095 (2017). arXiv:1710.02095 http://arxiv.org/abs/1710.02095

[16] Zhiheng Huang, Wei Xu, and Kai Yu. 2015. Bidirectional LSTM-CRF Models for Sequence Tagging. *CoRR* abs/1508.01991 (2015). arXiv:1508.01991 http://arxiv.org/abs/1508.01991

[17] Rada Mihalcea and Paul Tarau. 2004. TextRank: Bringing Order into Text. In *Proceedings of the 2004 Conference on Empirical Methods in Natural Language Processing , EMNLP 2004, A meeting of SIGDAT, a Special Interest Group of the ACL, held in conjunction with ACL 2004, 25-26 July 2004, Barcelona, Spain*. 404–411. http://www.aclweb.org/anthology/W04-3252

[18] Preslav Nakov, Doris Hoogeveen, Lluís Màrquez, Alessandro Moschitti, Hamdy Mubarak, Timothy Baldwin, and Karin Verspoor. 2017. SemEval-2017 Task 3: Community Question Answering. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval '17)*. Association for Computational Linguistics, Vancouver, Canada.

[19] Preslav Nakov, Lluís Màrquez, Walid Magdy, Alessandro Moschitti, Jim Glass, and Bilal Randeree. 2015. SemEval-2015 Task 3: Answer Selection in Community Question Answering. In *Proceedings of the 9th International Workshop on Semantic Evaluation (SemEval 2015)*. Association for Computational Linguistics, Denver, Colorado, 269–281. http://www.aclweb.org/anthology/S15-2047

[20] Preslav Nakov, Lluís Màrquez, Alessandro Moschitti, Walid Magdy, Hamdy Mubarak, Abed Alhakim Freihat, Jim Glass, and Bilal Randeree. 2016. SemEval-2016 Task 3: Community Question Answering. In *Proceedings of SemEval-2016*.

[21] Ramesh Nallapati, Bowen Zhou, Cícero Nogueira dos Santos, Çaglar Gülçehre, and Bing Xiang. 2016. Abstractive Text Summarization using Sequence-to-sequence RNNs and Beyond. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*, Yoav Goldberg and Stefan Riezler (Eds.). ACL, 280–290. http://aclweb.org/anthology/K/K16/K16-1028.pdf

[22] Ani Nenkova and Kathleen McKeown. 2012. A Survey of Text Summarization Techniques. In *Mining Text Data*, Charu C. Aggarwal, ChengXiang Zhai, and blubberdiblubb (Eds.). Springer, 43–76. http://dblp.uni-trier.de/db/books/collections/Mining2012.html#NenkovaM12

[23] Vinay Pande, Tanmoy Mukherjee, and Vasudeva Varma. 2013. Summarizing Answers for Community Question Answer Services. In *Language Processing and Knowledge in the Web - 25th International Conference, GSCL 2013, Darmstadt, Germany, September 25-27, 2013. Proceedings*. 151–161. https://doi.org/10.1007/978-3-642-40722-2_16

[24] Jeffrey Pennington, Richard Socher, and Christopher D Manning. 2014. Glove: Global Vectors for Word Representation.. In *EMNLP*, Vol. 14. 1532–1543.

[25] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 2383–2392. https://doi.org/10.18653/v1/D16-1264

[26] Abigail See, Peter J. Liu, and Christopher D. Manning. 2017. Get To The Point: Summarization with Pointer-Generator Networks. *CoRR* abs/1704.04368 (2017). arXiv:1704.04368 http://arxiv.org/abs/1704.04368

[27] Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to Rank Short Text Pairs with Convolutional Deep Neural Networks. In *Proceedings of the 38th International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR '15)*. ACM, New York, NY, USA, 373–382. https://doi.org/10.1145/2766462.2767738

[28] Anna Shtok, Gideon Dror, Yoelle Maarek, and Idan Szpektor. 2012. Learning from the past: answering new questions with past answers. In *Proceedings of the 21st international conference on World Wide Web*. ACM, 759–768.

[29] Hongya Song, Zhaochun Ren, Shangsong Liang, Piji Li, Jun Ma, and Maarten de Rijke. 2017. Summarizing Answers in Non-Factoid Community Question-Answering. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. 405–414. http://dl.acm.org/citation.cfm?id=3018704

[30] Chuanqi Tan, Furu Wei, Wenhui Wang, Weifeng Lv, and Ming Zhou. 2018. Multiway Attention Networks for Modeling Sentence Pairs. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence, IJCAI-18*. International Joint Conferences on Artificial Intelligence Organization, 4411–4417. https://doi.org/10.24963/ijcai.2018/613

[31] Ming Tan, Bing Xiang, and Bowen Zhou. 2015. LSTM-based Deep Learning Models for non-factoid answer selection. *CoRR* abs/1511.04108 (2015). arXiv:1511.04108 http://arxiv.org/abs/1511.04108

[32] Mattia Tomasoni and Minlie Huang. 2010. Metadata-Aware Measures for Answer Summarization in Community Question Answering. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*. 760–769. http://www.aclweb.org/anthology/P10-1078

[33] Xiaobing Xue, Jiwoon Jeon, and W Bruce Croft. 2008. Retrieval models for question and answer archives. In *Proceedings of the 31st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 475–482.