

CoSPLADE: Contextualizing SPLADE for Conversational Information Retrieval

Nam Le Hai¹[0000-0002-9020-8790], Thomas Gerald², Thibault Formal^{1,3},
Jian-Yun Nie⁴, Benjamin Piwowarski¹[0000-0001-6792-3262], and Laure
Soulier^{1,2}[0000-0001-9827-7400]

¹ Sorbonne Université, CNRS, ISIR, F-75005 Paris, France
first.last@sorbonne-universite.fr

² Université Paris-Saclay, CNRS, LISN, 91405 Orsay France *first.last@lisn.fr*

³ Naver Labs Europe, Meylan, France *first.last@naverlabs.com*

⁴ University of Montreal, Montreal, Canada *nie@iro.umontreal.ca*

Abstract. Conversational search is a difficult task as it aims at retrieving documents based not only on the current user query but also on the full conversation history. Most of the previous methods have focused on a multi-stage ranking approach relying on query reformulation, a critical intermediate step that might lead to a sub-optimal retrieval. Other approaches have tried to use a fully neural IR first-stage, but are either zero-shot or rely on full learning-to-rank based on a dataset with pseudo-labels. In this work, leveraging the CANARD dataset, we propose an innovative lightweight learning technique to train a first-stage ranker based on SPLADE. By relying on SPLADE sparse representations, we show that, when combined with a second-stage ranker based on T5Mono, the results are competitive on the TREC CAsT 2020 and 2021 tracks.

Keywords: information retrieval · conversational search · first-stage ranking.

1 Introduction

With the introduction of conversational assistants like Siri, Alexa or Cortana, conversational Information Retrieval, a variant of adhoc IR, has emerged as an important research domain [4,6]. In conversational IR, a search is conducted within a session, and the user’s information need is expressed through a sequence of queries, similarly to natural conversations – thus introducing complex interdependencies between queries and responses.

Not surprisingly, neural IR models have been shown to perform the best on conversational IR [5,7]. Most prior works rely on a Historical Query Expansion step [34], i.e. a query expansion mechanism that takes into account all past queries and their associated answers. Such query expansion model is learned on the CANARD dataset [8], which is composed of a series of questions and their associated answers, together with a disambiguated query, referred to as *gold query* in this paper. However, relying on a reformulation step is computationally

costly and might be sub-optimal as underlined in [13,16]. Krasakis et al. [13] proposed to use ColBERT [12] in a zero-shot manner, replacing the query by the sequence of queries, without any training of the model. Lin et al. [16] proposed to learn a dense *contextualized* representation of the query history, optimizing a learning-to-rank loss over a dataset composed of weak labels. This makes the training process complex (labels are not reliable) and long.

In this work, we follow this direction of research but propose a much lighter training process for the first-stage ranker, where we focus on queries and do not make use of any passage – and thus of a learning-to-rank training. It moreover sidesteps the problem of having to derive weak labels from the CANARD dataset. Given this strong supervision, we can consider more context – i.e. we use the answers provided by the system the user is interacting with, which allows to better contextualize the query, as shown in our experiments. The training loss we propose leverages the sparse representation of queries and documents provided by the SPLADE model [9]. In a nutshell, we require that the representation of the query matches that of the disambiguated query (i.e. the *gold query*). Our first-stage ranker achieves high performances, especially on recall – the most important measure in a multi-stage approach, comparable to the best systems in TREC CAsT [7], but also on precision-oriented measures – which shows the potential of our methodology.

Finally, to perform well, the second-stage ranker (i.e. re-ranker) needs to consider the conversation as well, which might require a set of heuristics to select some content and/or query reformulation such as those used in [18]. Leveraging the fact that our first-stage ranker outputs weights over the (BERT) vocabulary, we propose a simple mechanism that provides a conversational context to the re-ranker in the form of keywords selected by SPLADE.

In summary, our contributions are the following:

1. We propose a new loss to optimize a first-stage ranker resulting in a lightweight training strategy and state-of-the-art results in terms of recall;
2. We show that, when combined with a second-stage ranker based on a context derived from the SPLADE query representation of the first stage, we obtain results on par with the best approaches in TREC CAsT 2020 and 2021.

2 Related Works

The first edition [5] of the TREC Conversational Assistance Track (*CAsT*) was implemented in 2019, providing a new challenge on Conversational Search. The principle is the following: a user queries the system with questions in natural language, and each time gets a response from the system. The challenge differs from classical search systems as involving previous utterances (either queries or answers) is key to better comprehending the user intent. In conversational IR, and in TREC *CAsT* [6,5,7] in particular, the sheer size of the document collection implies to design an efficient (and effective) search system.

Conversational IR is closely related to conversational Question-Answering [25,27,26] in the sense that they both include interaction turns in natural language. However, the objective is intrinsically different. While the topic or the

context (i.e., the passage containing answers) is known in conversational QA, conversational IR aims to search among a huge collection of documents with potentially more exploratory topics. With this in mind, in the following, we focus on the literature review of conversational IR.

We can distinguish two lines of work in conversational search. The first one [29,30,32,3] focuses on a Contextual Query Reformulation (CQR) to produce a (plain or bag-of-words) query, representing ideally the information need free of context, which is fed into a search model. One strategy of CQR consists in selecting keywords from previous utterances by relying on a graph weighted by either word2vec similarity [29], term-based importance using BM25 [19], or classification models [30]. Other approaches [14,19,18,33,28] leverage the potential of generative language models (e.g., GPT2 or T5) to rewrite the query. Such approaches are particularly effective, reaching top performances in the TREC CAsT 2020 edition [5]. Query reformulation models also differ in the selected evidence sources. Models either focus on the early stage of the conversation [1], on a set of the queries filtered either heuristically [2] or by a classification model [21], or on both previous queries and documents [31]. Finally, to avoid the problem of generating a *single* query, [14,20] have proposed to use different generated queries and aggregate the returned documents.

The reformulation step is however a bottleneck since there is no guarantee that the “gold query” is optimal and thus generalizes well [16,13]. Moreover, generating text is time-consuming. To avoid these problems, the second line of work aims to directly integrate the conversation history into the retrieval model, bypassing the query reformulation step. As far as we know, only a few studies followed this path in conversational search. Qu et al [24] compute a query representation using the k last queries in the dialogue [15]. Similarly Lin et al. [16] average contextualized tokens embeddings over the whole query history. The representation is learned by optimizing a learning-to-rank loss over a collection with weak labels, which requires much care to ensure good generalization. Finally, Krasakis et al. [13] use a more lexical neural model, i.e. ColBERT [12], to encode the query with its context – but they do not finetune it at all. In this work, we go further by using a sparse model SPLADE [9], using a novel loss tailored to such sparse representations, and by using a lightweight training procedure that does not rely on passages, but only on a dataset containing reformulated queries.

3 Model

In TREC CAsT [5,7], each retrieval session contains around 10 turns of exchange. Each turn corresponds to a query and its associated canonical answer⁵ is provided as context for future queries. Let us now introduce some notations that we use to describe our model. For each turn $n \leq N$, where N is the last turn of the conversation, we denote by q_n and a_n respectively the corresponding query and its response. Finally, the context of a query q_n at turn n corresponds

⁵ Selected by the organizer as the most relevant answer of a baseline system.

to all the previous queries and answers, i.e. $q_1, a_1, q_2, a_2, \dots, q_{n-1}, a_{n-1}$. The main objective of the *TREC CAsT* challenges is to retrieve, for each query q_n and its context, the relevant passages.

In the next sections, we present our first-stage ranker and second-stage re-ranker, along with their training procedure, both based, directly or indirectly, on the SPLADE (v2) model described in [9]. SPLADE has shown results on par with dense approaches on in-domain collections while exhibiting stronger abilities to generalize in a zero-shot setting [9]. It outputs a sparse representation of a document or a query in the BERT vocabulary, which is key to our model during training and inference. The SPLADE model we use includes a contextual encoding function, followed by some aggregation steps: ReLU, log saturation, and max pooling over each token in the text. The output of SPLADE is a sparse vector with only positive or zero components in the BERT vocabulary space $\mathbb{R}^{|V|}$. In this work, we use several sets of parameters for the same SPLADE architecture and distinguish each version by its parameters θ , and the corresponding model by $SPLADE(\dots; \theta)$.

3.1 First stage

The original SPLADE model [9] scores a document using the dot product between the sparse representation of a document (\hat{d}) and of a query (\hat{q}):

$$s(\hat{q}, \hat{d}) = \hat{q} \cdot \hat{d} \quad (1)$$

In our work, like in [16], we suppose that the document representation has been sufficiently well-tuned on the standard ad-hoc IR task. The document embedding \hat{d} is thus obtained using the pre-trained SPLADE model, i.e. $\hat{d} = SPLADE([\text{CLS}] d; \theta_{SPLADE})$ where θ_{SPLADE} are the original SPLADE parameters obtained from HuggingFace⁶. These parameters are not fine-tuned during the training process. We can thus use standard indices built from the original SPLADE document representations to retrieve efficiently the top- k documents. In the following, we present how to contextualize the query representation using the conversation history. Then, we detail the training loss aiming at reducing the gap between the representation of the gold query and the contextualized representation.

Query representation. Like state-of-the-art approaches for first-stage conversational ranking [16,13], we contextualize the query with the previous ones. Going further, we propose to include the answers in the query representation process, which is easier to do thanks to our lightweight training.

To leverage both contexts, we use a simple model where the contextual query representation at turn n , denoted by $\hat{q}_{n,k}$, is the combination of two representations, $\hat{q}_n^{queries}$ which encodes the current query in the context of all the previous

⁶ The weights can be found at <https://huggingface.co/naver/splade-cocondenser-ensembledistil>

queries, and $\hat{q}_{n,k}^{answers}$ which encodes the current query in the context of k the past answers⁷. Formally, the contextualized query representation $\hat{q}_{n,k}$ is:

$$\hat{q}_{n,k} = \hat{q}_n^{queries} + \hat{q}_{n,k}^{answers} \quad (2)$$

where we use two versions of SPLADE parameterized by $\theta_{queries}$ for the full query history and $\theta_{answers,k}$ for the answers. These parameters are learned by optimizing the loss defined in Eq. (8).

Following [16], we define $\hat{q}_n^{queries}$ to be the query representation produced by encoding the concatenation of the current query and all the previous ones:

$$\hat{q}_n^{queries} = SPLADE([\text{CLS}] q_n [\text{SEP}] q_1 [\text{SEP}] \dots [\text{SEP}] q_{n-1}; \theta_{queries}) \quad (3)$$

using a set of specific parameters $\theta_{queries}$.

To take into account the answers that the user had access to, we need to include them in the representation. Following prior work [2], we can consider a various number of answers k , and in particular, we can either choose $k = 1$ (the last answer) or $k = n - 1$ (all the previous answers). Formally, the representation $\hat{q}_{n,k}^{answers}$ is computed as:

$$\hat{q}_{n,k}^{answers} = \frac{1}{k} \sum_{i=n-k}^{n-1} SPLADE(q_n [\text{SEP}] a_i; \theta_{answers,k}) \quad (4)$$

Training Based on the above, training aims at obtaining a good representation \hat{q}_n for the last issued query q_n , i.e. to contextualize q_n using the previous queries and answers. To do so, we can leverage the gold query q_n^* , that is, a (hopefully) contextualized and unambiguous query. We can compute the representation \hat{q}_n^* of this query by using the original SPLADE model, i.e.

$$\hat{q}_n^* = SPLADE(q_n^*; \theta_{SPLADE}) \quad (5)$$

For example, for a query "How old is he?" the matching gold query could be "How old is Obama?". The representation of the latter given by SPLADE would be as follows:

[("Obama", 1.5), ("Barack", 1.2), ("age", 1.2), ("old", 1.0), ("president", 0.8), ...]

where the terms "Obama" and "Barack" clearly appear alongside other words related to the current query ("old" and the semantically related "age").

We can now define the goal of the training, which is to reduce the difference between the gold query representation \hat{q}_n^* and the representation $\hat{q}_{n,k}$ computed by our model. An obvious choice of a loss function is to match the predicted and gold representations using cosine loss (since the ranking is invariant when scaling the query). However, as shown in the result section, we experimentally

⁷ In the experiments, we also explore an alternative model where answers and queries are considered at once.

found better results with a modified MSE loss, whose first component is the standard MSE loss:

$$Loss_{MSE}(\hat{q}_{n,k}, \hat{q}_n^*) = MSE(\hat{q}_{n,k}, \hat{q}_n^*) \quad (6)$$

In our experiments, we observed that models trained with the direct MSE do not capture well words from the context, especially for words from the answers. The reason is that the manually reformulated gold query usually only contains a few additional words from the previous turns that are directly implied by the last query. Other potentially useful words from the answers may not be included. This is a conservative expansion strategy which may not be the best example to follow by an automatic query rewriting process. We thus added an asymmetric MSE, designed to encourage term expansion from past answers, but avoid introducing noise by restricting the terms to those present in the gold query q_n^* . Formally, our asymmetric loss is:

$$Loss_{asym}(\hat{q}_{n,k}^{answers}, \hat{q}_n^*) = (\max(\hat{q}_n^* - \hat{q}_{n,k}^{answers}, 0))^2 \quad (7)$$

where the maximum is component-wise. This loss thus pushes the answer-biased representation $\hat{q}_{n,k}^{answers}$ to include tokens from the gold answer. Contrarily to MSE, it does not impose (directly) an upper bound on the components of the $\hat{q}_{n,k}^{answers}$ representation – this is done indirectly through the final loss function described below.

The final loss we optimize is a simple linear combination of the losses defined above, and only relies on computing two query representations:

$$Loss(\hat{q}_{n,k}, \hat{q}_n^*) = Loss_{MSE}(\hat{q}_{n,k}, \hat{q}_n^*) + Loss_{asym}(\hat{q}_{n,k}^{answers}, \hat{q}_n^*) \quad (8)$$

There is an interplay between the two components of the global loss. More precisely, $Loss_{asym}$ pushes the $\hat{q}_{n,k}^{answers}$ representation to match the golden query representation \hat{q}_n^* if it can, and $Loss_{MSE}$ pushes the queries-biased representation $\hat{q}_{n,k}$ to compensate if not. It thus puts a strong focus on extracting information from past answers, which is shown to be beneficial in our experiments.

Implementation details. For the first-stage, we initialize both encoders (one encoding the queries, and the other encoding the previous answer) with pre-trained weights from SPLADE model for adhoc retrieval. We use the ADAM optimizer with train batch size 16, learning rate 2e-5 for the first encoder and 3e-5 for the second. We fine-tune for only 1 epoch over the CANARD dataset.

3.2 Reranking

We perform reranking using a T5Mono [22] approach, where we enrich the raw query q_n with keywords identified by the first-stage ranker. Our motivation is that these words should capture the information needed to contextualize the raw query. The enriched query q_n^+ for conversational turn n is as follows:

$$q_n^+ = q_n. Context : q_1 q_2 \dots q_{n-1}. Keywords : w_1, w_2, \dots, w_K \quad (9)$$

where the w_i are the top- K most important words that we select by leveraging the first-stage ranker as follows. First, to reduce noise, we only consider words that appear either in any query q_i or in the associated answers a_i (for $i \leq n-1$). Second, we order words by using the maximum SPLADE weight over tokens that compose the word.⁸

We denote the T5 model fine-tuned for this input as $T5^+$. As in the original paper [22], the relevance score of a document d for the query q_n is the probability of generating the token “true” given a prompt $\text{pt}(q_n^+, d) = \text{“Query: } q_n^+. \text{ Document: } d. \text{ Relevant:”}$:

$$\text{score}(q_n^+, d; \theta) = \frac{p_{T5}(\text{true}|\text{pt}(q_n^+, d); \theta)}{p_{T5}(\text{true}|\text{pt}(q_n^+, d); \theta) + p_{T5}(\text{false}|\text{pt}(q_n^+, d); \theta)} \quad (10)$$

where θ are the parameters of the T5Mono model.

Differently to the first stage training, we fine-tune the ranker by aligning the scores of the documents, and not the weight of a query (which is obviously not possible with the T5 model). Here the “gold” score of a document is computed using the original T5Mono with the gold query q_n^* . The T5 model is initialized with weights made public by the original authors⁹, denoted as θ_{T5} . More precisely, we finetune the pre-trained T5Mono model using the MSE-Margin loss [11]. The loss function for the re-ranker (at conversation turn n , given documents d_1 and d_2) is calculated as follows:

$$\mathcal{L}_R = \left[(s(q_n^+, d_1; \theta_{T5+}) - s(q_n^+, d_2; \theta_{T5+})) - (s(q_n^*, d_1; \theta_{T5}) - s(q_n^*, d_2; \theta_{T5})) \right]^2$$

We optimize the θ_{T5+} parameters by keeping the original θ_{T5} to evaluate the score of gold queries.

Implementation details. We initialize θ_{T5+} as θ_{T5} , and fine-tune for 3 epochs, with a batch size of 8 and a learning rate 1e-4. We sample pairs (d_1, d_2) using the first-stage top-1000 documents: d_1 is sampled among the top-3, and d_2 among the remaining 997 to push the model to focus on important differences in scores.

4 Experimental Protocol

We designed the evaluation protocol to satisfy two main evaluation objectives: (i) Evaluating separately the effectiveness of the first-stage and the second-stage ranking components of our CoSPLADE model; (ii) Comparing the effectiveness of our CoSPLADE model with TREC CAsT 2020 and 2021 participants.

⁸ To improve coherence, we chose to make keywords follow their order of appearance in the context, but did not vary this experimental setting.

⁹ We used the Huggingface checkpoint <https://huggingface.co/castorini/monot5-base-msmarco>

4.1 Datasets

To train our model, we used the CANARD corpus¹⁰, a conversational dataset focusing on context-based query rewriting. More specifically, the CANARD dataset is a list of conversation histories, each being composed of a series of queries, short answers (human written) and reformulated queries (contextualised). The training, development, and test sets include respectively 31.538, 3.418, and 5.571 contextual and reformulated queries.

To evaluate our model, we used the TREC CAsT 2020 and 2021 datasets which include respectively 25 and 26 information needs (topics) and a document collection composed of the MS MARCO dataset, an updated dump of Wikipedia from the KILT benchmark, and the Washington Post V4 collection. For each topic, a conversation is available, alternating questions and responses (manually selected passages from the collection, aka canonical answers). For each question (216 and 239 in total), the dataset provides its manually rewritten form as well as a set of about 20 relevant documents. We use the former to define an upper-bound baseline (**Splade_GoldQuery**).

4.2 Metrics and baselines

We used the official evaluation metrics considered in the TREC CAsT 2020 and 2021, namely nDCG@3, MRR, Recall@X, MAP@X, nDCG@X, where the cut-off is set to 1000 for the CAsT 2020 and 500 for the CAsT 2021. For each metric, we calculate the mean and variance of performance across the different queries in the dataset. With this in mind, we present below the different baselines and scenarios used to evaluate each component of our model.

First-stage ranking scenarios. To evaluate the effectiveness of our first-stage ranking model (Section 3.1), we compare our approach CoSPLADE, based on the query representation of Eq. (2) with different variants (the document encoder is set to the original SPLADE encoder throughout our experiments): **SPLADE_rawQuery** (lower bound): SPLADE [10] using only the original and ambiguous user queries q_n ; **SPLADE_goldQuery** (kind of upper bound): SPLADE using the manually rewritten query q_n^* ; **CQE** [16], a state-of-the-art dense contextualized query representation learned using learning-to-rank on a dataset with pseudo-labels.

To model answers when representing the query using $\hat{q}_{n,k}^{answers}$, we used two historical ranges (“**All**” with $k = n - 1$ answers and “**Last**” where we use only the last one, i.e. $k = 1$) and three types of answer inputs: **Answer** in which answers are the canonical answers; **Answer-Short** in which sentences are filtered as in the best performing TREC CAsT approach [18]. This allows for consistent input length, at the expense of losing information; **Answer-Long** As answers from CANARD are short (a few sentences extracted from Wikipedia – contrarily to CAsT ones), we expand them to reduce the discrepancy between training and

¹⁰ <https://sites.google.com/view/qanta/projects/canard>

inference. For each sentence, we find the Wikipedia passage it appears in (if it exists in ORConvQA [23]), and sample a short snippet of 3 adjacent sentences from it.

Finally, we also conducted ablation studies (on the best of the above variants) by modifying either the way to use the historical context or the training loss: **flatContext** a one-encoder version of our SPLADE approach in which we concatenate all information of the context to apply SPLADE to obtain a single representation of the query (instead of two representations $\hat{q}_n^{queries}$ and $\hat{q}_{n,k}^{answers}$ as in Equations 2 and 3) trained using a MSE loss function (Eq. 6) since there is no more two representations. **MSE** the version of our SPLADE approach trained with the MSE loss (Eq. 6) instead of the proposed one (Eq. 8); **cosine** the version of our SPLADE approach trained with a cosine loss instead of the proposed loss (Eq. 8). The cosine loss is interesting because it is invariant to the scaling factor that preserves the document ordering (Eq.1).

Second-stage ranking scenarios. We consider different scenario for our second-stage ranking model: **T5Mono_RawQuery** the T5Mono ranking model [22] applied on raw queries; **T5Mono_GoldQuery** the T5Mono ranking model applied on gold queries; **T5Mono_CQR** the T5Mono ranking model applied on query reformulation generated with a pre-trained T5 (using the CANARD dataset); **CoSPLADE_[context]_[number]**: different versions of our second-stage ranking model input (Eq. 9), varying 1) the number K of keywords identified as relevant by the first-stage ranker: 5, 10, 20, and 2) the presence or absence of the past queries within the reformulation.

TREC participant baselines. For each evaluation campaign (2020 and 2021), we also compare our model with the best, the median and the lowest TREC CAsT participants presented in the two overviews [5,7], where participant are ranked according to the nDCG@3 metric.

5 Results

5.1 First-stage ranking effectiveness

In this section, we focus on the first-stage ranking component of our CoSPLADE model. To do so, we experiment different scenarios aiming at evaluating the impact of the designed loss (Eq. 8) and the modeling/utility of evidence sources (Equations 3 and 4). Results of these different baselines and scenarios on the TREC CAsT 2021 dataset are provided in Table 1. Similar trends are observed on CAsT 2020, but are not reported due to space limit.

In general, one can see that all variants of our approach (CoSPLADE_* models) outperform the scenario applying the initial version of SPLADE on raw and, more importantly, gold queries. This is very encouraging since this latter scenario might be considered an oracle, i.e. the query is manually disambiguated. Finally, we improve the results over CQE [16] for all the metrics – showing that

	Recall@500	MAP@500	MRR	nDCG@500	nDCG@3
Baselines					
SPLADE_rawQuery	30.8±2.7	5.5±0.9	21.3±2.9	17.8±1.8	13.1±2.1
SPLADE_goldQuery	68.8±2.0	16.1±1.2	55.5±3.3	42.8±1.7	38.3±2.8
CQE [17] from [7]	79.1	28.9	60.3	55.7	43.8
Effect of answer processing: CoSPLADE_...					
AllAnswers	79.5±2.2	28.8±1.7	61.7±3.1	55.3±2.0	46.5±2.9
AllAnswers-short	72.8±2.6	25.7±1.9	54.4±3.3	49.5±2.3	40.1±3.0
AllAnswers-long	80.4±2.1	29.3±1.8	62.0±3.2	55.6±2.1	48.9±3.0
LastAnswer	83.4±2.0	31.2±1.8	61.8±3.1	58.1±2.0	47.4±3.0
LastAnswer-short	79.2±2.2	28.1±1.8	61.4±3.3	54.3±2.1	46.4±3.0
LastAnswer-long	85.2±1.8	32.0±1.7	64.3±03.0	59.4±1.9	48.6±3.0
CoSPLADE_LastAnswer-long variants					
flatContext	77.0±2.0	26.0±2.0	55.0±3.0	52.0±2.0	42.0±3.0
MSE loss	70.9±2.4	21.6±1.7	48.7±3.4	45.2±2.3	39.6±3.1
cosine loss	70.4±2.5	22.6±1.7	52.5±3.3	46.9±2.2	39.0±3.0

Table 1. Effectiveness of different scenarios of our first-stage ranking model on the TREC CAsT 2021.

our simple learning mechanism, combined with SPLADE, allows for achieving SOTA performance.

Leveraging queries and answers history better contextualizes the current query. The results of the flatContext scenario w.r.t. to the SPLADE_goldQuery allows comparing the impact of evidence sources related to the conversation since they both use the same architecture (SPLADE). We can observe that it obtains better results than SPLADE_goldQuery (e.g., 77 vs. 68.8 for the Recall@500 metric), highlighting the usefulness of context to better understand the information need.

More detailed answers perform better. Since answers are more verbose than questions, including them is more complex, and we need to study the different possibilities (CoSPLADE_AllAnswers* and CoSPLADE_LastAnswer*). One can see that: 1) trimming answers (*-short) into a few keywords is less effective than considering canonical answers, but 2) it might be somehow effective when combined with the associated Wikipedia passage (*-long). Moreover, it seems more effective to consider only the last answer rather than the whole set of answers in the conversation history¹¹. Taking all together, these observations highlight the importance of the way to incorporate information from answers into the reformulation process.

Dual query representation with asymmetric loss leverages sparse query representations. The results of the flatContext scenario show that considering at once past queries and answers perform better (compared to the MSE loss scenario which is directly comparable). However, if we separate the representations *and*

¹¹ This might be due to the simple way to use past answers, i.e. Eq. 4, but all the other variations we tried did not perform better

	Recall@500	MAP@500	MRR	nDCG@500	nDCG@3
Baselines					
T5Mono_RawQuery	78.4±2.3	21.0±1.8	39.6±3.2	45.9±2.1	28.4±3.0
T5Mono_GoldQuery	86.1±1.7	44.1±1.9	78.7±2.7	68.5±1.8	64.6±2.8
T5Mono_CQR	80.4±2.2	30.0±1.9	58.2±3.4	55.3±2.1	44.6±3.2
coSPLADE-based second stage variants					
CoSPLADE_NoContext_5	84.3±1.8	31.7±2.0	61.6±3.3	58.1±2.0	45.9±3.1
CoSPLADE_NoContext_10	83.1±1.9	32.0±1.7	66.0±3.1	59.1±1.9	49.8±2.9
CoSPLADE_NoContext_20	84.8±1.7	33.4±1.8	66.0±3.0	60.4±1.8	49.6±2.9
CoSPLADE_Context_5	85.0±1.7	35.0±1.8	68.4±3.0	61.7±1.9	51.5±02.9
CoSPLADE_Context_10	84.8±1.7	36.5±1.9	67.8±3.1	63.0±1.9	53.3±3.1
CoSPLADE_Context_20	84.9±1.7	35.5±1.8	69.8±3.0	62.2±1.9	54.4±2.9

Table 2. Effectiveness of different scenarios of our second-stage ranking model on TREC CAsT 2021.

use an asymmetric loss function, the conclusion changes. Moreover, the comparison of our best scenario CoSPLADE_LastAnswer-long with a similar scenario trained by simply using a MSE or a cosine losses reveals the effectiveness of our asymmetric MSE (Equation 7). Remember that this asymmetric loss encourages the consideration of previous answers in the query encoding. This reinforces our intuition that the conversation context, and particularly verbose answers, is important for the conversational search task. It also reveals that the context should be included at different levels in the architecture (input and loss).

5.2 Second-stage ranking effectiveness

In this section, we rely on the CoSPLADE_LastAnswer-long model as a first stage ranker, and evaluate different variants of the second-stage ranking method relying on the T5Mono model. For fair comparison, we also mention results obtained by a T5Mono ranking model applied on raw and gold queries, as well as query reformulated using a T5 generative model. Results on the TREC CAsT 2021 dataset are presented in Table 2.

The analysis of the CoSPLADE model variants allows to highlight different observations regarding the usability of the context and the number of keywords added to the query. First, adding the previous questions to the current query in the prompt (i.e., “Context”) seems to improve the query understanding and, therefore, positively impacts the retrieval effectiveness. For instance, when 5 keywords are added, the context allows reaching 51.5% for the nDCG@3 against 45.9% without context. Second, the effectiveness metrics tend to increase with the number of additional keywords, particularly for scenarios without context, which is sensible. This trend is less noticeable for the scenarios with context since the best metrics are alternatively obtained by the scenario adding either 10 or 20 keywords. It is worth noting however that adding 10 or 20 keywords is more valuable than adding only 5 (e.g. 54.4% vs. 51.5% for the nDCG@3 metric). It thus seems that 1) keywords help to reformulate the initial information need,

2) but they can lead to saturation when they are both numerous and combined with other information.

By comparing the best model scenarios with the more basic scenarios applying the T5Mono second-stage ranker on raw and gold queries, we can observe that our method allows improving the retrieval effectiveness regarding initial queries but is not sufficient for reaching the performance of T5Mono_GoldQuery. However, results obtained when applying T5Mono on queries reformulated by T5 highlight that the contextualization of an initial query is a difficult task. Indeed, the T5Mono_CQR scenario is less effective than the T5Mono_GoldQuery one with between 6 and 20 points of difference depending on the metrics.

Moreover, it is interesting to note that the SPLADE model applied on raw and gold queries (first-stage ranking in Table 1) obtains lower results than the T5Mono model on the same data (second-stage ranking in Table 2). It can be explained by the purpose of those two architectures which are different: SPLADE is a sparse model focusing on query/document expansion while T5Mono is particularly devoted to increase precision. However, it is worth noting that combining SPLADE and T5Mono as first and second-stage rankers reaches the highest effectiveness results in our experimental evaluation. This shows the effectiveness of CoSPLADE to both contextualize queries and effectively rank documents.

5.3 Effectiveness compared to TREC CAsT participants

We finally compare our approach with TREC CAsT participants for the 2020 and 2021 evaluation campaigns. For both years, we can see that we obtain effectiveness metrics that are very close or higher than the ones reached by the best participants. Indeed, CoSPLADE surpasses the best TREC participant for the 2020 evaluation campaign regarding Recall@1000 and nDCG@1000. For 2021, our model obtains better results than the best one for the MRR and nDCG@3 metrics. For both years, the best participant is the h2olo team [18,7] where they use query reformulation techniques, either using AllenAI or T5. Our results suggest that our approach focusing on a sparse first-stage ranking model allows combining the benefit of query expansion and document ranking in a single model that eventually helps the final reranking step. In other words, simply rewriting the query without performing a joint learning document ranking can hinder the overall performance of the search task.

6 Conclusion

In this paper, we have shown how a sparse retrieval neural IR model, namely SPLADE [9], could be leveraged together with a lightweight learning process to obtain a state-of-the-art first-stage ranker. We further showed that this first-stage ranker could be used to provide context to the second-stage ranker, leading to results comparable with the best-performing systems. Future work may explore strategies to better capture the information from the context or to explicitly treat user feedback present in the evaluation dataset.

TREC CAsT 2020	Recall@1000	MAP@1000	MRR	nDCG@1000	nDCG@3
TREC Participant (best)	63.3	30.2	59.3	52.6	45.8
TREC Participant (median)	52.1	15.1	42.2	36.4	30.4
TREC Participant (low)	27.9	1.0	5.9	11.1	2.2
CoSPLADE	82.4±2.0	26.9±1.5	58.1±2.9	54.2±1.8	44.0±2.7
TREC CAsT 2021	Recall@500	MAP@500	MRR	nDCG@500	nDCG@3
TREC Participants 1 (best)	85.0	37.6	67.9	63.6	52.6
TREC Participants 2 (median)	36.4	17.6	53.4	33.6	37.7
TREC Participants 3 (low)	58.9	7.6	27.0	31.4	15.4
CoSPLADE	84.9±1.7	35.5±1.8	69.8±3	62.2±1.9	54.4±2.9

Table 3. TREC CAsT 2020 and 2021 performances regarding participants

References

- Aliannejadi, M., Chakraborty, M., Rissola, E.A., Crestani, F.: Harnessing evolution of multi-turn conversations for effective answer retrieval pp. 33–42. <https://doi.org/10.1145/3343413.3377968>, <http://arxiv.org/abs/1912.10554>
- Arabzadeh, N., Clarke, C.L.A.: Waterlooclarke at the trec 2020 conversational assistant track (2020)
- Clarke, C.L.A.: Waterlooclarke at the TREC 2019 conversational assistant track. In: Voorhees, E.M., Ellis, A. (eds.) Proceedings of the Twenty-Eighth Text REtrieval Conference, TREC 2019, Gaithersburg, Maryland, USA, November 13-15, 2019. NIST Special Publication, vol. 1250. National Institute of Standards and Technology (NIST) (2019), <https://trec.nist.gov/pubs/trec28/papers/WaterlooClarke.C.pdf>
- Culpepper, J.S., Diaz, F., Smucker, M.D.: Research frontiers in information retrieval: Report from the third strategic workshop on information retrieval in lorne (SWIRL 2018). SIGIR Forum **52**(1), 34–90 (2018). <https://doi.org/10.1145/3274784.3274788>, <https://doi.org/10.1145/3274784.3274788>
- Dalton, J., Xiong, C., Callan, J.: CAsT 2020: The conversational assistance track overview p. 10
- Dalton, J., Xiong, C., Callan, J.: TREC CAsT 2019: The conversational assistance track overview <http://arxiv.org/abs/2003.13624>
- Dalton, J., Xiong, C., Callan, J.: TREC CAsT 2021: The Conversational Assistance Track Overview p. 7 (2021)
- Elgohary, A., Peskov, D., Boyd-Graber, J.: Can You Unpack That? Learning to Rewrite Questions-in-Context. In: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP). pp. 5918–5924. Association for Computational Linguistics, Hong Kong, China (Nov 2019). <https://doi.org/10.18653/v1/D19-1605>, <https://aclanthology.org/D19-1605>
- Formal, T., Lassance, C., Piwowarski, B., Clinchant, S.: From Distillation to Hard Negative Sampling: Making Sparse Neural IR Models More Effective. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2353–2359. SIGIR ’22, Association for Computing Machinery, New

- York, NY, USA (Jul 2022). <https://doi.org/10.1145/3477495.3531857>, <http://doi.org/10.1145/3477495.3531857>
10. Formal, T., Piwowarski, B., Clinchant, S.: SPLADE: Sparse Lexical and Expansion Model for First Stage Ranking. In: Proceedings of the 44th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 2288–2292. SIGIR '21, Association for Computing Machinery, New York, NY, USA (Jul 2021). <https://doi.org/10/gm2tf2>, <https://doi.org/10.1145/3404835.3463098>
 11. Hofstätter, S., Althammer, S., Schröder, M., Sertkan, M., Hanbury, A.: Improving efficient neural ranking models with cross-architecture knowledge distillation. ArXiv **abs/2010.02666** (2020)
 12. Khattab, O., Zaharia, M.: ColBERT: Efficient and effective passage search via contextualized late interaction over BERT <http://arxiv.org/abs/2004.12832>
 13. Krasakis, A.M., Yates, A., Kanoulas, E.: Zero-shot Query Contextualization for Conversational Search. In: Proceedings of the 45th International ACM SIGIR Conference on Research and Development in Information Retrieval. pp. 1880–1884. SIGIR '22, Association for Computing Machinery, New York, NY, USA (Jul 2022). <https://doi.org/10.1145/3477495.3531769>, <https://doi.org/10.1145/3477495.3531769>
 14. Kumar, V., Callan, J.: Making information seeking easier: An improved pipeline for conversational search p. 10
 15. Lan, Z., Chen, M., Goodman, S., Gimpel, K., Sharma, P., Soricut, R.: ALBERT: A lite BERT for self-supervised learning of language representations. In: 8th International Conference on Learning Representations, ICLR 2020, Addis Ababa, Ethiopia, April 26-30, 2020. OpenReview.net (2020), <https://openreview.net/forum?id=H1eA7AEtvS>
 16. Lin, S.C., Yang, J.H., Lin, J.: Contextualized query embeddings for conversational search <http://arxiv.org/abs/2104.08707>
 17. Lin, S.C., Yang, J.H., Lin, J.: In-batch negatives for knowledge distillation with tightly-coupled teachers for dense retrieval. In: Proceedings of the 6th Workshop on Representation Learning for NLP (RepL4NLP-2021). pp. 163–173. Association for Computational Linguistics. <https://doi.org/10.18653/v1/2021.repl4nlp-1.17>, <https://aclanthology.org/2021.repl4nlp-1.17>
 18. Lin, S.C., Yang, J.H., Lin, J.: TREC 2020 Notebook: CAsT Track. Tech. rep., TREC (Dec 2021)
 19. Lin, S.C., Yang, J.H., Nogueira, R., Tsai, M.F., Wang, C.J., Lin, J.: Multi-stage conversational passage retrieval: An approach to fusing term importance estimation and neural query rewriting <http://arxiv.org/abs/2005.02230>
 20. Lin, S., Yang, J., Nogueira, R., Tsai, M., Wang, C., Lin, J.: Query reformulation using query history for passage retrieval in conversational search. CoRR **abs/2005.02230** (2020), <https://arxiv.org/abs/2005.02230>
 21. Mele, I., Muntean, C.I., Nardini, F.M., Perego, R., Tonello, N.: Finding Context through Utterance Dependencies in Search Conversations. Tech. rep. (2021)
 22. Nogueira, R., Jiang, Z., Pradeep, R., Lin, J.: Document ranking with a pre-trained sequence-to-sequence model. In: Findings of the Association for Computational Linguistics: EMNLP 2020. pp. 708–718. Association for Computational Linguistics. <https://doi.org/10.18653/v1/2020.findings-emnlp.63>, <https://www.aclweb.org/anthology/2020.findings-emnlp.63>

23. Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W.B., Iyyer, M.: Open-retrieval conversational question answering pp. 539–548. <https://doi.org/10.1145/3397271.3401110>, <http://arxiv.org/abs/2005.11364>
24. Qu, C., Yang, L., Chen, C., Qiu, M., Croft, W.B., Iyyer, M.: Open-retrieval conversational question answering. In: Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 539–548. SIGIR '20, Association for Computing Machinery, New York, NY, USA (2020). <https://doi.org/10.1145/3397271.3401110>, <https://doi.org/10.1145/3397271.3401110>
25. Qu, C., Yang, L., Qiu, M., Croft, W.B., Zhang, Y., Iyyer, M.: Bert with history answer embedding for conversational question answering. In: Proceedings of the 42nd International ACM SIGIR Conference on Research and Development in Information Retrieval. p. 1133–1136. SIGIR'19, Association for Computing Machinery, New York, NY, USA (2019). <https://doi.org/10.1145/3331184.3331341>, <https://doi.org/10.1145/3331184.3331341>
26. Qu, C., Yang, L., Qiu, M., Zhang, Y., Chen, C., Croft, W.B., Iyyer, M.: Attentive history selection for conversational question answering. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management. pp. 1391–1400 (2019)
27. Reddy, S., Chen, D., Manning, C.D.: CoQA: A conversational question answering challenge. Transactions of the Association for Computational Linguistics **7**, 249–266 (2019). https://doi.org/10.1162/tacl_a_00266, <https://aclanthology.org/Q19-1016>
28. Vakulenko, S., Longpre, S., Tu, Z., Anantha, R.: Question rewriting for conversational question answering. In: Proceedings of the 14th ACM International Conference on Web Search and Data Mining. pp. 355–363. ACM. <https://doi.org/10.1145/3437963.3441748>, <https://dl.acm.org/doi/10.1145/3437963.3441748>
29. Voskarides, N., Li, D., Panteli, A., Ren, P.: ILPS at TREC 2019 conversational assistant track p. 4
30. Voskarides, N., Li, D., Ren, P., Kanoulas, E., de Rijke, M.: Query resolution for conversational search with limited supervision pp. 921–930. <https://doi.org/10.1145/3397271.3401130>, <http://arxiv.org/abs/2005.11723>
31. Yan, X., Clarke, C.L.A., Arabzadeh, N.: Waterlooclarke at the trec 2021 conversational assistant track (2021)
32. Yang, J.H., Lin, S.C., Wang, C.J., Lin, J.J., Tsai, M.F.: Query and answer expansion from conversation history. In: TREC (2019)
33. Yu, S., Liu, J., Yang, J., Xiong, C., Bennett, P., Gao, J., Liu, Z.: Few-shot generative conversational query rewriting <http://arxiv.org/abs/2006.05009>
34. Zamani, H., Trippas, J.R., Dalton, J., Radlinski, F.: Conversational Information Seeking (Jan 2022). <https://doi.org/10.48550/arXiv.2201.08808>, <http://arxiv.org/abs/2201.08808>, arXiv:2201.08808 [cs]