ELSEVIER

Since January 2020 Elsevier has created a COVID-19 resource centre with free information in English and Mandarin on the novel coronavirus COVID-19. The COVID-19 resource centre is hosted on Elsevier Connect, the company's public news and information website.

Elsevier hereby grants permission to make all its COVID-19-related research that is available on the COVID-19 resource centre - including this research content - immediately available in PubMed Central and other publicly funded repositories, such as the WHO COVID database with rights for unrestricted research re-use and analyses in any form or by any means with acknowledgement of the original source. These permissions are granted for free by Elsevier for as long as the COVID-19 resource centre remains active.

Knowledge-Based Systems 240 (2022) 108072

Contents lists available at ScienceDirect

Knowledge-Based Systems

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

Information retrieval and question answering: A case study on COVID-19 scientific literature

Arantxa Otegi [a], Iñaki San Vicente [b,*], Xabier Saralegi [b], Anselmo Peñas [c], Borja Lozano [c], Eneko Agirre [a]

[a] HiTZ Center - Ixa, UPV/EHU, Spain
[b] Elhuyar fundazioa, Spain
[c] NLP & IR Group, UNED, C/Juan del Rosal 16, 28040 Madrid, Spain

ARTICLE INFO

Article history:
Received 5 August 2021
Received in revised form 21 December 2021
Accepted 24 December 2021
Available online 31 December 2021

MSC:
00-01
99-00

Keywords:
Question answering
COVID-19

ABSTRACT

Biosanitary experts around the world are directing their efforts towards the study of COVID-19. This effort generates a large volume of scientific publications at a speed that makes the effective acquisition of new knowledge difficult. Therefore, Information Systems are needed to assist biosanitary experts in accessing, consulting and analyzing these publications. In this work we develop a study of the variables involved in the development of a Question Answering system that receives a set of questions asked by experts about the disease COVID-19 and its causal virus SARS-CoV-2, and provides a ranked list of expert-level answers to each question. In particular, we address the interrelation of the Information Retrieval and the Answer Extraction steps. We found that a recall based document retrieval that leaves to a neural answer extraction module the scanning of the whole documents to find the best answer is a better strategy than relying in a precise passage retrieval before extracting the answer span.

© 2021 Published by Elsevier B.V.

1. Introduction

Many bio-sanitary researchers around the world are directing their efforts towards the study of COVID-19. This effort generates a large volume of scientific publications and at a speed that makes the effective acquisition of new knowledge difficult. Information Systems are needed to assist bio-sanitary experts in accessing, consulting and analyzing these publications.

The ultimate goal of this research is to develop a system that receives a set of questions asked by experts about the disease COVID-19 and its causal virus SARS-CoV-2, and returns a ranked list of expert-level answers to each question, extracted from scientific literature as collected in the CORD-19 document collection about COVID-19 [1].

Given the size of the document collection (over 400,000 articles), it is customary that, for each given question, to first apply Information Retrieval (IR) to retrieve the most relevant contexts (documents or passages), and then extract the answer from those contexts using a neural Question Answering (QA) system.[1]

In the COVID-19 domain, answers are long and they have multiple dimensions (nuggets) that must be returned to provide a complete correct answer. Ours is a general scenario where the different nuggets of relevant information can come from different documents and, therefore, the system must avoid to return irrelevant or repeated information. Solving this task requires a three step architecture. First an initial retrieval of contexts (documents or passages) where the candidate nuggets may appear. Second, the selection of text spans out of these contexts containing the relevant nuggets. And third, the ranking of these text spans. The evaluation measures will evaluate the quality of this ranking promoting relevant information and avoiding irrelevant or repeated nuggets.

Table 1 shows an example of the task. In this example, the system has returned four different contexts. These contexts can come from the same or different documents. Then, for each context, the system has selected the text spans (marked in bold face) that will be evaluated against the list of expected relevant nuggets. In general, the system will consider not only four contexts, but hundreds or thousands so, after this process, the system must provide the best possible ranking of all text spans coming from all different retrieved contexts. The evaluation of this ranking will consider the coverage of the expected nuggets.

This three-step architecture raises several research questions:

1. Related to the system architecture, which is the best strategy: a system relying on a precise passage retrieval before

* Corresponding author.
E-mail addresses: arantza.otegi@ehu.eus (A. Otegi), i.sanvicenteg@elhuyar.eus (I. San Vicente), x.saralegi@elhuyar.eus (X. Saralegi), anselmo@lsi.uned.es (A. Peñas), blozano@lsi.uned.es (B. Lozano), e.agirre@ehu.eus (E. Agirre).

[1] This QA architecture has been named recently as retriever–reader. In this work we will refer indistinctly the reader as Answer Extraction step since the task is to identify inside a context the span that gives answer to the question.

https://doi.org/10.1016/j.knosys.2021.108072
0950-7051/© 2021 Published by Elsevier B.V.

Table 1
Sample of question and answers in COVID-19 domain.

Question	What is the origin of COVID 19?
Expected nuggets	'spillover', 'positive selection pressure', 'the species barrier', 'potential mutations', 'genetic recombination', 'animal-to-human transmission', 'bat reservoirs', 'Codon usage bias', 'ancestral haplotypes', 'bat coronavirus genome', 'zoonotic origin', 'seafood wholesale market in Wuhan', 'evolutionary constraints', 'pangolins', 'interspecies transmission', 'betacoronaviruses', 'viral fitness', 'molecular evolution', 'Chinese province of Hubei', 'Mammal species', 'Bats', 'virus adapation', 'species of origin', 'emergence'
Retrieved contexts and text spans	
1	**It is improbable that SARS-CoV-2 emerged through laboratory manipulation** of a related SARS-CoV-like coronavirus. As noted above, the RBD of SARS-CoV-2 is optimized for binding to human ACE2 with an efficient solution different from those previously predicted.
2	Furthermore, if genetic manipulation had been performed, one of the several reverse-genetic systems available for betacoronaviruses would probably have been used.
3	However, the genetic data irrefutably show that **SARS-CoV-2 is not derived from any previously used virus backbone**.
4	Instead, we propose **two scenarios that can plausibly explain the origin of SARS-CoV-2: (i) natural selection in an animal host before zoonotic transfer; and (ii) natural selection in humans following zoonotic transfer**.

extracting the answer span or a recall based document retrieval leaving to the Answer Extraction module the scanning of documents to evaluate the passages and find the best answer?

2. Related to the IR step, which is the best way to compose a query? How can the IR module be tuned for the COVID domain? Given that the retrieval is a previous step for Answer Extraction, must it be optimized for ranking quality based on Normalized Discounted Cumulative Gain (NDCG) or recall? Since the Answer Extraction module considers paragraph-size contexts, is indexing at passage level better than indexing at document level?
3. Related to the Answer Extraction step, how can the QA module be tuned for the COVID domain and specially to the type of long answers that the task requires? How to produce the final ranking of answers considering both IR and QA scores?

To answer these questions we have conducted a series of experiments taking advantage of the Epidemic Question Answering (EPIC-QA) dataset.[2]

The main contribution of this work is the experimentation that gives answers to the above research questions and the proposal of a system architecture following those answers that returns a ranked list of expert-level answers to questions related to COVID-19.

2. Previous work

Open-domain QA aims to answer questions by finding answers in a large collection of documents [2]. Early approaches to solve this problem consisted in elaborated systems with multiple components [3,4]. Recent advances follow a two-step pipeline, the *retriever–reader* [5]. The retriever first extracts a small subset of contexts from a large collection. This component is most commonly approached using an ad-hoc IR engine, but over the past years, alternative neural architectures have been proposed [6–12]. Among the proposed approaches, those based on pre-trained language models stand out, such as [11,12], as they offer a significant improvement over classic term-matching based IR systems. Those approaches use the neural model to rerank an initial ranking generated by a classical IR model based on term-matching techniques. [11] propose a neural reranker based on BERT Large to address the task of passage retrieval. Specifically, they fine-tune the BERT Large model for the task of binary classification, adding a single layer neural network fed by the [CLS] vector for the purpose of obtaining the probability of the passage being relevant. These probabilities are then used to rank the final relevant passages. [12] adopt a similar strategy to address the document retrieval task. Because documents exceed the maximum length of BERT's input, they divide the documents into sentences, and add their scores. A known issue of such neural architectures is that they require a large number of query relevances (qrels) for training, but their manual generation is very expensive. Some authors [11,12] use qrel data oriented to passage retrieval such as MS-Marco [13] and TREC-CAR [14]. Another alternative is to generate relevance judgments automatically. [15], for example, propose to train neural models for ranking using pseudo-qrels generated by unsupervised models like BM25. The TREC-CAR dataset [14] itself is automatically generated from the structure (article, section and paragraph) of the Wikipedia articles. [16] generate pseudo-qrels from a news collection, using the titles as pseudo-queries and their content as relevant text.

The second component of the pipeline, the reader (or Answer Extraction module), scans each context thoughtfully in search for a span that contains the answer to the question. [5] encode the retrieved contexts and the questions using different recurrent neural networks. For each question-context pair, two distributions over the contexts tokens are computed using bilinear terms, one for the start of the span and the other for the end. The final answer maximizes the probability of the start and end tokens. With the advent of transformers and pre-trained language models many systems adopted them as their reader [17]. These systems, although effective at extracting correct answers from a context, process each question-context pair as independent of each other. To improve on this issue [18] normalizes the probabilities of the span start and end for all tokens in all contexts whereas [19] adds another distribution over the [CLS] token representation of all contexts. Other approaches substitute the reader by an answer reranking module [11,20] where the retrieved passages are divided into plausible sentences which are used as the span of the answer. These sentences are then further reranked by a cross-encoder.

Recently some authors proposed generative models that generate the answer instead of extracting it [21]. Although competitive in some benchmarks, large generative models are expensive to train and make inferences on. To tackle this problem [22] combine evidence from the retrieved passages to generate the answer. Note also that some systems use symbolic knowledge to complement the background knowledge in pre-trained transformers [23,24]. The symbolic knowledge has been shown to be useful in tasks such as OK-VQA [25] where the answer is not contained in the target document, and background knowledge is needed in order to be able to answer.

[2] https://bionlp.nlm.nih.gov/epic_qa/.

One crucial part of the pipeline is the *granularity* of the passages that the retriever extracts for the reader to scan. Early works studied the downstream effect of this parameter in the retriever with [26] suggesting full documents might lead to better QA performance whereas [27] conclude that small passages with high coverage allow a smaller search space for the QA system to find the correct answer. Most recent work is inconclusive about which type of textual length (full documents [5], natural passages [28] or sentences [29]) works best.

With the rise of the COVID-19 Pandemic the value of open-domain QA systems increased as the academic literature about the virus became unmanageable. Many systems, like "CORD-19 Search" by Vespa[3], "@Cord-19 Search" by AWS[4] [30], "COVID-19 Research Explorer" by Google[5] [31] or "Covidex" by Waterloo University and New York University[6] [32] arose during the first months of the pandemic. Albeit useful in aiding scientific search of COVID-19 literature they all lacked proper domain evaluation, which is usually performed by comparing the correct span of text with the predicted one using a set metric like *F1* or an *Exact Match* [33]. This evaluation is well suited for short and factoid answers but fails to capture complex responses to diverse information needs within the same question. Previous evaluation scenarios introduced the concept of *nugget* (as atomic information pieces to be recovered) and differentiated between "vital" nuggets and "non-vital" nuggets [34].

In order to check our research questions we take advantage of a recent evaluation proposal (EPIC-QA) [35] which includes the search of relevant documents and the extraction of the answer from those documents. Note that there are other evaluation datasets for QA about COVID-19 [36,37], but they provide the target document containing the answer, and as such, are not useful to answer our research questions. EPIC-QA is relevant for complex QA scenarios that combines ranking metrics such as NDCG with *nuggets* to provide a new evaluation metric called Normalized Discount Novelty Score (NDNS, discussed later).

The best performing systems in EPIC-QA are based on a two-stage pipeline which includes a retriever and an answer extraction module. [38] return full sentences as answers, and thus they use two reranker language models for scoring the sentences, returning the top sentence as answer. [39] also return sentences, using the ROUGE score to filter sentences in their ranked set. [40] use BERT-based to rerank and a generative transformer for filtering. All these systems retrieve paragraphs instead of documents, and do not explore one of our research question: why to retrieve paragraphs instead of documents, which allows the reader to scan larger contexts?

While convolutional and recurrent neural networks have been used in the past [41,42], the current state-of-the-art relies heavily on transformer neural networks [43] which are often pre-trained using different variants of language model losses [17,44]. Transformers have been applied to natural language processing discriminative classifiers, but recent trends have also used generative models with success [21,45]. Pre-trained models are based on large quantities of text, and some models have explored hybrid architectures which tap the semantic information in knowledge graphs [46]. Current neural models for QA demand large amounts of training data. There are some attempts to generalize the learning from fewer data points. For example, [47] explore the extension of existing capsule networks into a new framework with advantages concerning scalability, reliability and generalizability, showing promising results in QA. In this work we have focused on the use of pre-trained discriminative transformer models [17].

3. Architecture overview and research questions

The proposed system has an architecture with three steps: context retrieval (documents or passages); context scanning for answer extraction; and ranking of answers. Each of these steps requires some experimentation before we can conclude about the best way to adapt them to the COVID-19 domain, as set out in the introduction.

3.1. Context retrieval

Our IR module follows two main steps, preliminary retrieval and reranking. Before indexing the collection, a keyword-based filter is applied to select only COVID related documents, since CORD-19 also includes papers focused on other coronaviruses. Keywords are different variants of the "COVID-19" term, which are used to filter out up to 37.5% of the documents. Previous experiments done for the TREC-COVID challenge [48] showed the effectiveness of this filtering for improving the retrieval (see Section 4.1).

Related to retrieval, there is a research question about which strategy is best, a fine-grained passage retrieval before extracting the answer span, or a document based retrieval leaving to the Answer Extraction module the scanning of full documents to evaluate the passages and find the best answer. For this reason, we will conduct our experiments on the whole architecture for both options, and see which is the most appropriate at the end.

Regarding preliminary retrieval, we obtain an initial ranking for the query from the collection of full texts of the scientific articles. We use a language modeling based IR approach [49] including Pseudo Relevance Feedback (PRF). For that purpose, we used the Indri search engine [50], which combines Bayesian networks with language models. The query and documents are tokenized and stemmed using Krovetz stemmer [51], and stopwords are removed.

The adaptation of this system to the COVID-19 domain requires some experimentation that will be addressed in Section 5.1. First, the EPIC-QA questions have three fields (keywords based query, natural language question and narrative or background). Thus, there is a question about how we should construct the query to best exploit the information contained in those fields.

Second, there is a question about the number of contexts (passages or documents) to retrieve before feeding the QA module. That is, find the balance between the recall of the retrieval and the noise that the QA module can manage.

Regarding reranking, the preliminary ranking obtained in the previous step is reranked using a BERT-based relevance classifier, following a strategy similar to the one proposed by [11]. For each candidate document given by the preliminary ranking, its abstract and the corresponding query are processed through a BERT-based relevance classifier, which returns a probability of an abstract to be relevant with respect to the given query. Section 5.1.2 gives further details on the experimentation done on this regard.

3.2. Context scanning and answer extraction

The answer extraction module is based on neural network techniques. More specifically, we have used the SciBERT language representation model, which is a pre-trained language model based on BERT, but trained on a large corpus of scientific text, including text from the biomedical domain [52]. SciBERT was selected for this module over other language models adapted to the biomedical domain (e.g. BioBERT [53], Clinical BERT [54]) based on the results obtained in initial experiments for EPIC-QA participation.

[3] https://cord19.vespa.ai/.

[4] https://cord19.aws/.

[5] https://covid19-research-explorer.appspot.com/.

[6] https://covidex.ai/.

We fined-tuned SciBERT for QA using SQuAD2.0 [33], which is a reading comprehension dataset widely used in the QA research community. Following the usual answer extraction method [33] we used this fined-tuned SciBERT model as a pointer network, which selects an answer start and end index given a question and a context. According to the EPIC-QA guidelines, the answers returned by the QA system must be a sentence or several contiguous sentences. In our case, we select those sentences which contain the answer span delimited by the start and end indexes given by the neural network.

In case the input contexts exceed the maximum input sequence length (e.g. when working with full documents) we follow the sliding window approach where the documents are split into overlapping passages. For the maximum sequence length, stride parameters and other parameters we used the default values of [55].

After scanning the whole context (passage or document depending on the strategy), we keep the most probable answers to the question for each. So at this step, there are several research questions we must address to adapt the system to the COVID-19 domain as follows.

First, about the best dataset to fine-tune the SciBERT model for the target task. SQuAD 2.0 aims at relatively short factoid questions, while in this dataset, questions are complex and answers are expected to be longer. Therefore, we need to assess our hypothesis that using QuAC [56] in addition to SQuAD 2.0 when fine-tuning SciBERT will improve system results, as QuAC is a conversational QA dataset containing a higher rate of non-factoid questions than SQuAD.

Second, we need to determine both the appropriate number of relevant contexts that will be scanned by the answer extraction module, and, the number of candidate answers that will be extracted from each context. The idea is to find a balance between different answers that come from different documents and those that are in a single document, without introducing to much noise when producing the final ranking of the answers per each question.

Third, we have to find out whether we will consider each context corresponding to the same question as independent from each other when normalizing the scores of the answers extracted from them. Considering contexts independently could originate incomparable answer scores if these answers come from different contexts. Thus we will explore if normalizing globally the scores across all relevant contexts for each question is helpful or not.

Finally, the last question to address is which will be the exact question given as an input to the answer extraction module. Each topic of the EPIC-QA dataset provides three different fields as it will be described in Section 4.1. We need to figure out if using the text provided in the question field, which is how humans post a question using natural language, is enough to get the correct answers, or if some other piece of information provided in other fields is needed (for example, the more elaborated information provided in background field).

The first three questions will be addressed in Section 5.2.1 by an extensive hyperparameter exploration, whereas the last question will be answered once all other hyperparameters are fixed in Section 5.2.3.

3.3. Ranking of answers

At this point, each answer comes with two relevance evidences: the context retrieval score, and the score given by the answer extraction from the context. Therefore, we need to study which is the best way to combine both evidences and produce the final ranking of answers. We will focus on this issue together with other questions formulated in the previous section in the hyperparameter exploration carried out in Section 5.2.1.

4. Evaluation setting

To the best of our knowledge, there is only one dataset aimed at the evaluation of complete QA systems related to COVID-19: EPIC-QA. In this section we describe this dataset together with the evaluation measures used for our study.

4.1. Datasets

CORD-19[1] is a resource of over 400,000 scholarly articles, including over 150,000 with full text, about COVID-19, SARS-CoV-2, and related coronaviruses. The CORD-19 dataset represents the most extensive machine-readable coronavirus literature collection. It is used extensively for research, including international shared tasks in the IR and QA fields, such as the CORD-19 Challenge at Kaggle,[7] TREC-COVID [57] or EPIC-QA.[8]

EPIC-QA track aims to develop systems capable of automatically answering ad-hoc questions about COVID-19. EPIC-QA involves two tasks, Expert QA and Consumer QA. Experiments in this work are conducted with the data related to the Expert QA task, aimed to answer questions posed by experts.

The questions have three fields: a keyword-based query, a natural language question, and a narrative or background. They are evaluated through the use of *nuggets*, a set of atomic "facts" that answer the question. Two datasets were compiled for the task:

The Preliminary Round dataset uses a snapshot of CORD-19 from June 19, 2020, and it includes 45 expert level questions used in the 4th round of the TREC-COVID IR shared task. EPIC-QA organizers annotated human-generated answers and sentence-level answer annotations (judgments for short) for 21 of those questions as evaluation set in the preliminary round. All development experiments in this work (see Section 5) are carried out using this dataset.

The Primary Round dataset is compiled using a snapshot of CORD-19 from October 22, 2020, and it includes 30 expert level questions and their respective relevance judgments. We use this dataset to evaluate our final systems in Section 6.

In addition to EPIC-QA datasets, the CORD-19 version used in the final round of the TREC-COVID shared task[9] and the associated document level relevance judgments are used to fine-tune the reranker module of the IR engine responsible for the preliminary retrieval. See Section 5.1.2 for details. The dataset contains 192 K scientific articles, and relevance judgments for 50 topics.

4.2. IR evaluation

In order to evaluate our IR systems two well known evaluation measures were selected, both used also in the TREC-COVID [57] shared task, specifically NDCG and recall at different cutoffs of the ranking.

NDCG is a measure of ranking quality widely used to evaluate search engine results. Roughly, it takes into account both the order of the results (more relevant results should be on top, if not NDCG penalizes those results) and also different lengths of result rankings. For us, it gives a measure of how good is the ranking of document/passages we provided to the answer extraction module. The higher the NDCG, the less noise should

[7] https://www.kaggle.com/allen-institute-for-ai/CORD-19-research-challenge.

[8] https://bionlp.nlm.nih.gov/epic_qa/.

[9] Release of July 16, 2020.

Table 2
IR Results on epic-qa-dev regarding the fields used as a query: (i) query; (ii) query+question: query and question concatenated; and (iii) w(qry+qs)+(1 − w)backg: complex query built concatenating query and question fields, and combining linearly the concatenation with the background field.

(a) IR Results on epic-qa-dev for **passage retrieval** over 5000 element rankings with different query building strategies.

Query building	NDCG	R@500	R@1K	R@2K	R@3K	R@4K	R@5K
query	0.2977	0.2864	0.3766	0.4606	0.5397	0.5689	0.5918
query+question	0.3832	0.3906	0.4898	0.5848	0.6599	0.7101	0.7346
0.5 ·(qry+qs) + 0.5 ·backg	0.3901	0.3979	0.5126	0.6099	0.6795	0.7154	0.7604

(b) IR Results on epic-qa-dev for **document retrieval** over 5000 element rankings with different query building strategies.

Query building	NDCG	R@500	R@1K	R@2K	R@3K	R@4K	R@5K
query	0.4758	0.5548	0.6518	0.6951	0.7308	0.7494	0.7602
query+question	0.5555	0.6478	0.7492	0.8102	0.8382	0.8566	0.8606
0.7 ·(qry+qs) + 0.3 ·backg	0.5613	0.6636	0.7575	0.8222	0.8437	0.8589	0.8625

the answer extraction module handle. It is computed as follows for a ranking of p elements:

$$NDCG_p = \frac{DCG_p}{IDCG_p}$$

where DCG is $DCG_p = \sum_{i=1}^{p} \frac{rel_i}{\log_2(i+1)}$ (1)

being rel_i the relevance value of the ith element in the ranking,
and $IDCG$ is the DCG that an ideal ranking would have at
position p

Recall is the fraction of the documents that are relevant to the query (TP+FN) that are successfully retrieved (TP). We compute recall at various cutoffs, for both passage and document retrieval. Recall gives a measure of how many good candidates the IR engine retrieves and hands to the answer extraction module regardless of the rank. The higher the recall, the higher the number of good results the answer extraction module can retrieve. Recall is computed as follows:

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

where TP = True Positives and FN = True negatives.

4.3. Scenarios and evaluation measures for the complete system

The main evaluation metric is NDNS, which was provided in the EPIC-QA track, and is a modified version of NDCG, where relevance is given by a set of gold nuggets that the answer should contain. Given a list of ranked answers for a question the *Novelty Score* (NS) measures the relevant information (as given in the gold nuggets) not yet seen in previous answers higher in the ranked list, as follows:

$$NS(a) = \frac{n_a \cdot (n_a + 1)}{n_a + f_a} \tag{3}$$

where n_a is the number of novel nuggets of answer a and f_a is the *sentence factor* which weights the score based on the number of sentences in a. Three different variants of NDNS are considered based on how this factor is computed:

- **Exact**: Answers should express novel nuggets in as few sentences as possible. This scenario is more suited to evaluate systems where brevity is a priority, like a chat bot which can only give one answer. The sentence factor is the number of sentences ($n_{sentences}$) in the answer:

$$f_a = n_{\text{sentences}} = n_{\text{non-relevant}} + n_{\text{redundant}} + n_{\text{novel}} \tag{4}$$

  where $n_{non-relevant}$ is the number of sentences with no nuggets, $n_{redundant}$ is the number of sentences that contain previously seen nuggets and n_{novel} is the number of sentences with novel nuggets.

- **Relaxed**: Length does not penalize answers as long as every sentence contains novel nuggets. This variant of the NDNS metric rewards systems where brevity is not a requirement but non-redundancy is.

$$f_a = n_{\text{non-relevant}} + n_{\text{redundant}} + \min(n_{\text{novel}}, 1) \tag{5}$$

- **Partial**: Redundant information is not penalized which makes this metric well suited for systems solving tasks like a state of the art research about a topic where some overlap in the relevant answers is expected.

$$f_a = n_{\text{non-relevant}} + \min(n_{\text{novel}}, 1) \tag{6}$$

The final metric is computed as the cumulative NS of answers up to rank $k = 1000$:

$$NDNS(\mathbf{a}) = \frac{1}{NDNS_{\text{ideal}}} \cdot \sum_{r=1}^{k} \frac{NS(a_r)}{\log_2(r+1)} \tag{7}$$

where $NDNS_{\text{ideal}}$ is the optimal ranking of answers that could have been found in the document collection for the given question, computed using a beam-search with a width of 10 over the annotated sentences.

5. Experimentation

5.1. Information retrieval

As mentioned in Section 3.1 our IR module follows a two step approach [58]: preliminary retrieval and re-reranking.

With respect to the evaluation of the IR module, the gold-standard associated with the EPIC-QA dataset contains nuggets annotated at sentence level. In order to evaluate the IR systems at passage and document level, we have created qrels at these two levels, annotating as relevant passages or documents those that contain at least one relevant nugget.

5.1.1. Field used as query for IR

The first aspect we have explored is how we will construct the query we send to the IR engine, using the fields available in the topics. For these first experiments PRF values are set with default values ($fbt = 50$ and $fbd = 10$), and no reranking is applied.

Table 2 presents the results of those experiments. Results are given in terms of NCDG and recall at different cutoffs. We analyzed various field concatenations. In order to combine query, question and background fields, we also experimented with constructing complex queries, assigning different weights to query, question and narrative (background) fields. Extensive experiments were conducted for field combinations, but for the sake of readability, we only show the linear combination that obtained the best results in last rows of Table 2a and b. Weights were optimized for each setup. Linear combinations were also tested

Table 3
IR Results on epic-qa-dev. Ranking quality and recall oriented systems are evaluated with and without neural reranking, for both passage and document retrieval strategies. RR column (2nd) indicates if reranking is used or not and when used the value of k.

(a) IR Results on epic-qa-dev for **passage retrieval** using neural reranking over 5000 element ranking. Recall oriented (PRF $fbt = 125$ $fbd = 30$) vs. ranking quality (NDCG) oriented (PRF $fbt = 125$, $fbd = 10$) systems.

PRF optimized for	RR	NDCG	R@500	R@1K	R@2K	R@3K	R@4K	R@5K
NDCG	no	0.3993	0.3991	0.5169	0.6252	0.6796	0.728	0.7599
R	no	0.3855	0.4002	0.5059	0.6118	0.6842	0.7253	0.7665
NDCG	0.9	0.4157	0.4612	0.5787	0.6724	0.7238	0.7604	0.76
R	0.9	0.403	0.456	0.5665	0.6705	0.7195	0.7555	0.76288

(b) IR Results on epic-qa-dev for **document retrieval** using neural reranking over 5000 element ranking. Recall oriented (PRF $fbt = 30$, $fbd = 40$) vs. ranking quality (NDCG) oriented (PRF $fbt = 10$, $fbd = 40$) systems.

PRF optimized for	RR	NDCG	R@500	R@1K	R@2K	R@3K	R@4K	R@5K
NDCG	no	0.5781	0.6731	0.761	0.8214	0.8363	0.8487	0.8652
R	no	0.5596	0.6651	0.7675	0.8255	0.8446	0.8598	0.8712
NDCG	0.1	0.5807	0.7041	0.7867	0.8342	0.8538	0.8655	0.8686
R	0.2	0.5691	0.7041	0.7661	0.8271	0.8538	0.8655	0.8709

for the *query+question* case, but concatenation yielded better results.

Results show that using complex queries perform best, both at passage level (see Table 2a) and at document level (see Table 2b). Hence we adopted the following query building strategy for the rest of the IR experiments:

$$Q = w \cdot (query + question) + (1 - w) \cdot background$$
$$\text{where } w = 0.5 \text{ for passage retrieval and } w = 0.7 \text{ for document retrieval.} \tag{8}$$

5.1.2. Neural reranking

As we have already mentioned, the preliminary ranking obtained in the previous step is reranked using a BERT-based relevance classifier, following a strategy similar to the one proposed by [11]. In the case of document retrieval, for each candidate document given by the preliminary ranking, its abstract and the corresponding query are processed through a BERT-based relevance classifier, which returns a probability of an abstract to be relevant with respect to the given query. In the case of passage retrieval the candidate passage is processed with the corresponding query.

For this purpose, we fine-tuned the Clinical BERT pre-trained model [54] on the task of identifying relevant abstracts with respect to queries. Clinical BERT is trained on top of BioBERT [53] using clinical notes. If we compare Clinical BERT with SciBERT, which is used for answer extraction, both are trained on biomedical domain data. SciBERT contains data from Semantic Scholar,[10] while Clinical BERT/BioBERT include data from PubMed[11] and PMC.[12] We selected Clinical BERT for our context retrieval system based on the results obtained on the TREC-COVID dataset [58]. In order to train the neural reranker a set of queries and their respective relevant and non-relevant documents are needed. The objective is to learn the classification – the relevance of the second text with respect to the first – of a pair of texts. We use two different query relevance sets to fine-tune our reranker:

- We exploit the title–abstract relationship [58]. Titles of scientific articles are usually brief and at the same time descriptive of the content. Therefore, they are similar to the queries used in search systems, and can be used as a pseudo-query. Its corresponding abstract constitutes a good candidate to be a relevant text (pseudo-positive) to that pseudo-query. We take (title,abstract) pairs to generate (pseudo-query,pseudo-positive) pairs. Non-relevant (pseudo-negative) texts are generated by randomly selecting abstracts ($n = 2$) from the collection. The CORD-19 version used in the final round of the TREC-COVID shared task[13] was used to automatically generate this training dataset. This dataset contains 369,930 title–abstract pair relevance judgments.
- TREC-COVID shared task official query relevance set, comprising 69,316 query–abstract pair judgments.

Fine-tuning was done in two steps, first over the automatically generated pseudo-qrel dataset and then over the TREC-COVID relevance judgments dataset. All fine-tunings were performed using original BERT Tensorflow implementation on Google cloud V3-8 TPUs. Training was done for 4 epochs with a learning rate of 2e-5 and a batch size of 32.

As mentioned, the classifier returns a relevance probability of an abstract with respect to a given query. This probability is linearly combined with the score of the first ranking according to a coefficient k, and the ranking is rearranged based on that new value. Eq. (9) shows the linear combination formula. Neural reranking k was optimized with the EPIC-QA preliminary round collection.

$$score_d = k \cdot (RerankerScore_d) + (1 - k) \cdot (IndriScore_d) \tag{9}$$

Table 3 presents the results of the experiments carried out with and without reranking. Reranking weight (k value in Eq. (9)) was optimized for each setup ([0..1], 0.1 intervals). PRF is applied optimizing the number of documents and the number of terms (fbt) with respect to NDCG and recall metrics, looking for a ranking that is either quality oriented or recall oriented, respectively. Thus, for both passages and documents, two PRF setups were tested, optimized for NDCG and recall, respectively (Table 3 captions report the respective parameter values).

Regarding passages (see Table 3a), ranking quality oriented systems not only outperform recall oriented systems on NDCG, but they also show very competitive recall performances, even outperforming recall oriented systems for some cutoffs. The same trend is observed for document retrieval (see Table 3b), where ranking quality oriented systems are again better in terms of NDCG and they are on par with recall oriented systems in terms of recall.

[10] https://www.semanticscholar.org/.

[11] https://pubmed.ncbi.nlm.nih.gov/.

[12] https://www.ncbi.nlm.nih.gov/pmc/.

[13] Release of July 16, 2020.

Reranking improves results for both passage and document retrieval, and the trend observed in favor of ranking quality oriented systems is more accentuated for systems using reranking, being superior to recall oriented systems in all but recall@5K cutoff.

With those results in hand, quality oriented ranking settings and the use of reranking are selected for the remaining experiments.

5.1.3. Passages vs. documents

There is a final question about retrieval regarding the granularity of the textual fragments to be handed to the answer extraction module: passage retrieval so the answer is extracted directly from the passage, or document retrieval so the answer extraction module has to scan the full document in order to select both the passage and the answer.

We carried out three sets of experiments to find out which IR engine (passage or document) would offer the best starting point to the answer extraction module in terms of recall of documents, passages and nuggets, respectively.

In order to measure the recall at document level, passage-based retrieval rankings must be converted to document rankings. In order to do so, passages are substituted by their corresponding document, and duplicates are removed from the ranking, i.e., a document is given the rank of its top ranked passage. Results in Table 4a show that documents offer a better recall if we were to give the first 500 elements in the rank, but otherwise passages would be preferable.

To be fair with both strategies, recall at passage level should also be measured. In order to convert document rankings to passage rankings, documents are expanded inserting all the passages of a document in the ranking position of the document. This leads to very large rankings because documents contain 9.7 passages on average. In order to compare passage rankings with similar sizes, 5000 document rankings are retrieved and expanded, and they are compared to 50,000 retrieved passages rankings.[14] As expected, the document to passage conversion leads to a decay in the recall in the top part of the ranking (see Table 4b which smooths only when using very large rankings. As in the document level evaluation retrieving passages is the best performing strategy.

EPIC-QA has the concept of nuggets, which introduce the factor of finding not only relevant information, but also "new" information. The third experiment measures the recall of the IR systems in terms of the nuggets retrieved. Table 4c presents the results for different ranking cuttoffs. The same document expansion strategy as in the previous experiment is used to compare document and passage retrieval performance. Passages have the upperhand, due to the fact that passage ranking have more diverse answers and thus a bigger chance to find new nuggets in earlier positions of the rank.

Lastly, up until now, we have evaluated the performance of the IR systems using IR measures. But what if we were evaluating the output of the IR systems directly as answers to the EPIC-QA questions? In order to do that, we prepared 3 systems, to check whether the conclusions would be the same:

- **pas**: full retrieved passages are returned from the first context to the last.
- **doc2pas**: documents are expanded to passages as done for the previous recall experiments, and then full passages are returned as answer candidates.
- **pas-to-sent**: Instead of returning full passages, the first sentence of each candidate in the passage ranking is returned.

Table 4d presents the results using EPIC-QA metrics. Returning full passages (pas system) obtains the best results in terms of NDNS partial and relaxed metrics, but performs poorly on NDNS Exact metric as expected, since this last metric penalizes returning incorrect spans. In turn, pas-to-sent, returning shorter spans, performs significantly better on NDNS Exact than the others. We can see that the more elements we evaluate in the rankings the better the results. Note that official EPIC-QA evaluation only took into account the first 1000 candidates. Regarding passages vs. docs, EPIC-QA evaluation is inline with IR results, with passage retrieval clearly outperforming document retrieval. Thus, according to the IR step alone, passages would seem to be a better starting point for the answer extraction module.

5.2. Context scanning for answer extraction

In this section we first check the hyperparameters, then the linear combination of retrieval and answer scores, and finally explore the most appropriate field to use as question.

5.2.1. Hyperparameter exploration

In order to answer the research questions regarding context scanning for the answer extraction module discussed in Section 3.2, we carried out an exploration of all hyperparameters in question of the module. Two independent explorations have been carried out: the first one for the system that its context retrieval module is based on passages, and the second one for the system that uses documents as contexts. For each hyperparameter tuning, we consider the following hyperparameters: *fine-tuning dataset*, *number of contexts*, *number of answers*, *normalization* and *score combination*. Next, we will give more details about them.

The hyperparameter *fine-tuning dataset* considers which is the best dataset to fine-tune the SciBERT model for QA: only SQuAD dataset, or both, SQuAD and QuAC datasets. The *number of contexts* hyperparameter fixes how many contexts per each question will be scanned by this module. The values considered for it in this exploration are 1000, 5000 and 10,000 for the passage-based system, and 100, 500 and 1000 for the system based on documents. The hyperparameter related to the *number of answers* determines the number of candidate answers that each context will provide, and the values we explored are the following: 1, 2, 3, 5, 10 and 15. The *normalization* hyperparameter is related to the softmax normalization that is computed as a last step in the neural network of SciBERT. In this neural network two softmax classifiers are used to get two probability distributions over all tokens of a candidate answer span. The token with the highest probability according to the first classifier is selected as the start token of the candidate answer span. Similarly, the end token of the span is selected according to the second classifier. This normalization can be computed at context-level or collection-level, and these are the two possible values for this hyperparameter. In the former case, only spans from one context are taken into account when computing the probabilities, whereas all spans from all relevant contexts are considered in the latter. The last hyperparameter, *score combination*, is used to set the best way to combine the two scores obtained by each of the modules of the system, context retrieval module and answer extraction module, and these are the four possible combinations we have considered: the sum of both scores (ir+qa), the product of both scores (ir·qa), the sum of both scores but after applying the z-score normalization[15] to both scores (ir_norm+qa_norm) and the product of both z-scores (ir_norm·qa_norm).

The aim of this exploration is to find the best combination of hyperparameter values. Although all three variants of the NDNS

[14] 5000 docs $\times 10$ passages per doc $= 50{,}000$ passages.

[15] z-score is computed as $X_{normalized} = \frac{X - X_{mean}}{X_{stddev}}$.

Table 4
IR Results on epic-qa-dev for Passage vs. Document retrieval experiments.

(a) IR Results on epic-qa-dev for **document recall**: Passages vs. Documents. 5000 passages ranking vs. 5000 document rankings.

Index	reranking	Test EPIC-QA_docs					
		R@500	R@1K	R@2K	R@3K	R@4K	R@5K
passages	yes	0.6959	0.7979	0.8597	0.8693	0.8716	0.8716
documents	yes	0.7041	0.7867	0.8342	0.8538	0.8655	0.8686

(b) IR Results on epic-qa-dev for **passage recall**: Passages vs. Documents. 50,000 passages ranking vs. 5000 document ranking.

Index	Test EPIC-QA_passages								
	R@100	R@500	R@1K	R@2K	R@5K	R@1K0	R@15K	R@30K	R@50K
pas	0.2199	0.4645	0.582	0.6815	0.7577	0.8321	0.8676	0.9027	0.9082
docs	0.0867	0.2234	0.3336	0.4345	0.6189	0.7442	0.7883	0.8422	0.8724

(c) IR Results on epic-qa-dev for **nugget recall**: Passages vs. Documents. 50,000 passages ranking vs. 5000 document ranking.

Index	Test EPIC-QA_passages								
	N@100	N@500	N@1K	N@2K	N@5K	N@10K	N@15K	N@30K	N@50K
pas	0.5988	0.7934	0.88	0.9134	0.941	0.9589	0.9626	0.9774	0.9774
docs	0.3068	0.5156	0.6537	0.758	0.8661	0.921	0.942	0.9629	0.9644

(d) **EPIC-QA evaluation results** on epic-qa-dev: Passages vs. Documents. **EPIC-QA metrics**. 50,000 passages ranking vs. 5000 document ranking.

NDNS@k	Test EPIC-QA - epicQA evaluation								
	pas			doc2pas			pas2sent		
	Partial	Relaxed	Exact	Partial	Relaxed	Exact	Partial	Relaxed	Exact
1,000	0.2463	0.2296	0.1823	0.1124	0.1031	0.0936	0.1786	0.1799	0.2072
5,000	0.2519	0.2347	0.1871	0.1299	0.1180	0.1084	0.2098	0.2113	0.2425
15,000	0.2541	0.2362	0.1886	0.1338	0.1213	0.1118	0.2250	0.2266	0.2601
30,000	0.2552	0.2371	0.1893	0.1357	0.1228	0.1133	0.2274	0.2290	0.2628

Table 5
Best values of the hyperparameters for the passage- and document-based systems, and their results.

Context	Passage	Document
fine-tune	squad+quac	squad+quac
number_contexts	1,000	100
number_answers	15	15
normalization	document-level	document-level
combination	ir_norm+qa_norm	ir_norm+qa_norm
NDNS-Partial	0.2178	0.3044
NDNS-Relaxed	0.2177	0.3051
NDNS-Exact	0.2482	0.3411

have high correlation with each other we have chosen to focus on *Relaxed* as in the biomedical research shortness of relevant responses is not a requirement, but non-redundancy is. Table 5 shows the hyperparameter values of the best combination to obtain the maximum NDNS-Relaxed score on the preliminary round dataset, for both passage- and document-based systems. The results obtained using the evaluation metrics described in Section 4.3 are also shown in the table. Interestingly, the hyperparameter values for both systems are the same, except the number of contexts (1000 passages vs. 100 documents), which are equivalent as the average number of passages in a document is around 10. This exploration revealed that the best strategy to extract the answer from a document is to index and retrieve the whole document (and not specific passages), as the document-based system clearly outperforms the passage-based system.

For an in-depth analysis of the hyperparameter values Fig. 1 illustrates some results of the hyperparameter exploration for both passage- and document-based systems. For each hyperparameter, we show the maximum result obtained by the system when a hyperparameter is fixed to each of its values. Regarding to the dataset used for fine-tuning, using QuAC in addition to SQuAD clearly improves results for the document-based system (see Fig. 1(a)), which is what we expected as questions are complex and answers are expected to be longer than for factoid questions. According to Fig. 1(b), performance decreases when more contexts are scanned by the answer extraction module. The number of candidate answers provided by the answer extraction module does not affect significantly the performance of the passage-based system, but the results are higher when more answers are provided by the document-based system (Fig. 1(c)). Fig. 1(d) shows that document-level normalization is a better choice for both systems. Finally, we can see in Fig. 1(e) that applying z-score normalization to the linear combination of both scores is the best.

5.2.2. Exploration of the optimum weight for linear combination

As explained above, the best way to combine the score given by the context retrieval module and the score given by the answer extraction module to produce the final ranking is the linear combination of both scores, but after applying the z-score normalization. In the above exploration both scores were weighted equally, but we wanted to explore the best value for the weight (k) in the linear combination:

$$final\_score = (k \cdot cr\_score) + ((1 - k) \cdot ae\_score)$$

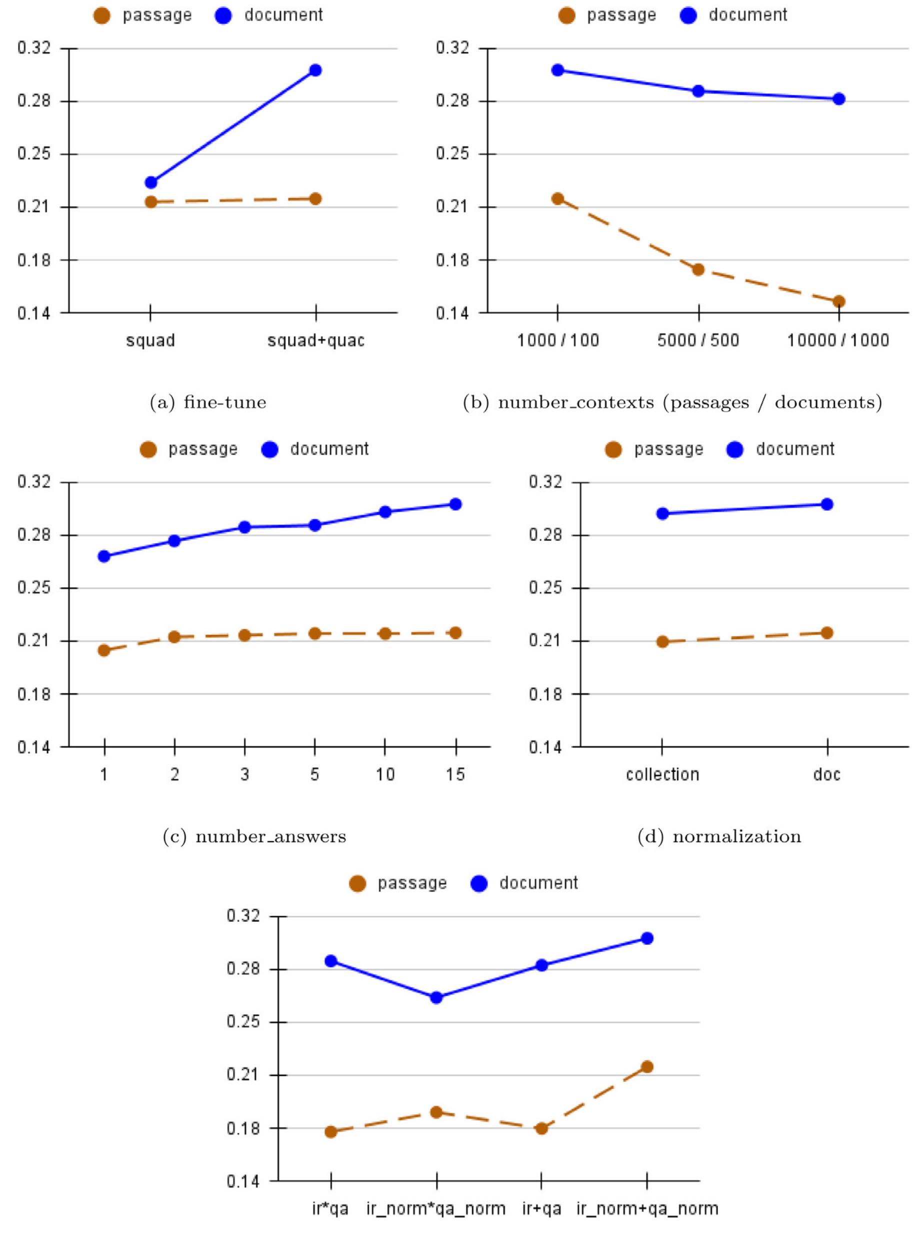
where *cr_score* is the score given by the context retrieval module and *ae_score* is the score given by the answer extraction module.

Fig. 2 shows the NDNS-Relaxed results for different k values, and its optimum value is 0.5 for both systems. We fixed all the hyperparameters of both systems at their best values as shown in Table 5 for this exploration.

5.2.3. Exploration of the most appropriate field to use as question

In this section we want to explore which field of the question (query, question, background or a combination of some of these) we should use to get the best performance of the answer extraction module. Note that we have used the text in the “question” field as a query in all the explorations we have carried out in the previous sections. For this exploration we fixed the best hyperparameter values (see Table 5) and we set $k = 0.5$ for the linear combination.

The results obtained in this exploration for both passage- and document-based systems can be seen in Table 6. Passage-based



(a) fine-tune

(b) number_contexts (passages / documents)

(c) number_answers

(d) normalization

(e) combination

Fig. 1. NDNS-Relaxed results (y-axis) of the exploration for each of the values of the hyperparameters.

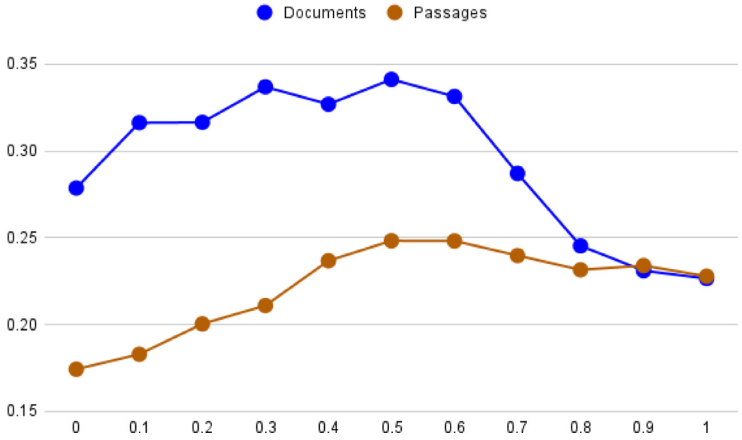


Fig. 2. NDNS-Relaxed results for different values of *k* in linear combination.

Table 6
Results of the exploration of which text field to use as a question.

Passages-based system			
Question field	NDNS-Partial	NDNS-Relaxed	NDNS-Exact
query	0.1938	0.1946	0.2225
question	0.2178	0.2177	0.2482
background	0.2147	0.2160	0.2474
query+question	**0.2200**	**0.2203**	**0.2510**
question+background	0.2173	0.2176	0.2477
Document-based system			
Question field	NDNS-Partial	NDNS-Relaxed	NDNS-Exact
query	0.2188	0.2195	0.2489
question	**0.3044**	**0.3051**	**0.3411**
background	0.2538	0.2554	0.2837
query+question	0.2833	0.2844	0.3195
question+background	0.2652	0.2659	0.2965

Table 7
Results on Primary dataset for passages-based and document-based full QA systems.

System	NDNS-Partial	NDNS-Relaxed	NDNS-Exact
Passages	0.2200	0.2196	0.2487
Documents	0.2860	0.2860	0.3241

system performs best when using the concatenation of query and question fields, while using only the question field obtains the best results for the document-based system. Therefore, adding extra information from the background field to the input does not yield better performance in any case.

6. Test on EPIC-QA primary dataset

The study performed so far run over the Preliminary dataset of EPIC-QA. The conclusion at this point is that a recall based document retrieval that leaves to a neural answer extraction module the scanning of whole documents to find the best answer is a better strategy than relying in a precise passage retrieval before extracting the answer span.

We wanted to check this result over the Primary dataset, which was unseen during the whole development process described above. Results are shown in Table 7 and they confirm the previous observation.

The results shows that in all scenarios, the performance of the document-based system is better than the passage-based one.

7. Conclusions

In this paper we have analyzed how to construct a system that extracts answers about questions on COVID from the scientific literature. We have performed extensive experiments to check which is the most effective combination of the retrieval and answer extraction components.

If we pay attention to IR results with IR metrics, results suggest that passage retrieval offers a better starting point for the QA module that extracts the actual answer. However, when we take into account the QA metrics, results show that document retrieval clearly outperforms passage retrieval. To obtain this result, the system must use smaller document rankings (around 500 candidates), and the neural QA module for extracting the answer must be fine-tuned properly.

At this respect, using QuAC dataset for additional fine-tuning after SQuAD over a SciBERT model showed the best results. EPIC-QA questions are complex and usually require longer answers beyond the factoid-like questions that are more common in other datasets like SQuAD. The additional fine-tuning with QuAC helped us to overcome this issue.

Our experiments also showed that adding the extra information in the task query description (background or narrative fields) when posing the questions is useful in the IR module, but is not effective in the QA module.

Finally, the ranking of answers for a given question is more effective if it combines both the relevance scores from the retrieval engine and scores for the extracted answer span. In our case, we obtain the best results giving the same weight to each evidence in a linear combination.

CRediT authorship contribution statement

Arantxa Otegi: Methodology, Software, Investigation. **Iñaki San Vicente:** Methodology, Software, Investigation. **Xabier Saralegi:** Conceptualization, Methodology, Software, Investigation. **Anselmo Peñas:** Conceptualization, Writing – original draft, Writing – review & editing. **Borja Lozano:** Software, Validation. **Eneko Agirre:** Supervision, Writing – review & editing.

Declaration of competing interest

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

Acknowledgments

This work has been partially funded by VIGICOVID project.[16] FSuperaCovid-5 (Fondo Supera COVID-19/CRUE-CSIC-Santander), and is partially supported via the IARPA BETTER Program, Spain contract No. 2019-19051600006 (ODNI, IARPA), by the Basque Government, Spain (IXA excellence research group IT1343-19 and DeepText KK-2020/00088) and by the Spanish Ministry of Science, Innovation and Universities (DeepReading RTI2018-096846-B-C21, MCIU/ AEI/FEDER, UE). We also acknowledge the support of Googles's TFRC program.

References

[1] L.L. Wang, K. Lo, Y. Chandrasekhar, R. Reas, J. Yang, D. Burdick, D. Eide, K. Funk, Y. Katsis, R.M. Kinney, et al., CORD-19: The COVID-19 Open Research Dataset, in: Proceedings of the 1st Workshop on NLP for COVID-19 at ACL 2020, 2020.
[2] E.M. Voorhees, et al., The TREC-8 Question Answering Track Report, in: Proceedings of the 8th Text REtrieval Conference, TREC-8, 1999, pp. 77–82.
[3] E. Brill, S. Dumais, M. Banko, An analysis of the AskMSR question-answering system, in: Proceedings of the 2002 Conference on Empirical Methods in Natural Language Processing, EMNLP 2002, 2002, pp. 257–264, http://dx.doi.org/10.3115/1118693.1118726.

[16] http://nlpuned.es/vigicovid-project.

[4] D.A. Ferrucci, Introduction to this is Watson, IBM J. Res. Dev. 56 (3.4) (2012) 235–249, http://dx.doi.org/10.1147/JRD.2012.2184356.

[5] D. Chen, A. Fisch, J. Weston, A. Bordes, Reading wikipedia to answer open-domain questions, in: Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, Volume 1: Long Papers, 2017, pp. 1870–1879, http://dx.doi.org/10.18653/v1/P17-1171.

[6] J. Guo, Y. Fan, Q. Ai, W.B. Croft, A deep relevance matching model for ad-hoc retrieval, in: Proceedings of the 25th ACM International on Conference on Information and Knowledge Management, 2016, pp. 55–64, http://dx.doi.org/10.1145/2983323.2983769.

[7] C. Xiong, Z. Dai, J. Callan, Z. Liu, R. Power, End-to-end neural ad-hoc ranking with kernel pooling, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 55–64, http://dx.doi.org/10.1145/3077136.3080809.

[8] Z. Dai, C. Xiong, J. Callan, Z. Liu, Convolutional neural networks for soft-matching N-grams in ad-hoc search, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018, pp. 126–134, http://dx.doi.org/10.1145/3159652.3159659.

[9] K. Hui, A. Yates, K. Berberich, G. De Melo, Co-PACRR: A context-aware neural IR model for ad-hoc retrieval, in: Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining, 2018, pp. 279–287, http://dx.doi.org/10.1145/3159652.3159689.

[10] B. Mitra, F. Diaz, N. Craswell, Learning to match using local and distributed representations of text for web search, in: Proceedings of the 26th International Conference on World Wide Web, 2017, pp. 1129–1299, http://dx.doi.org/10.1145/3038912.3052579.

[11] R. Nogueira, K. Cho, Passage Re-ranking with BERT, 2019, arXiv preprint arXiv:1901.04085.

[12] W. Yang, H. Zhang, J. Lin, Simple applications of BERT for ad hoc document retrieval, 2019, arXiv:1903.10972, CoRR abs/1903.10972.

[13] P. Bajaj, D. Campos, N. Craswell, L. Deng, J. Gao, X. Liu, R. Majumder, A. McNamara, B. Mitra, T. Nguyen, M. Rosenberg, X. Song, A. Stoica, S. Tiwary, T. Wang, MS MARCO: A human generated machine reading comprehension dataset, 2018, arXiv:1611.09268, CoRR abs/1611.09268.

[14] L. Dietz, M. Verma, F. Radlinski, N. Craswell, Trec complex answer retrieval overview, in: TREC, 2017.

[15] M. Dehghani, H. Zamani, A. Severyn, J. Kamps, W.B. Croft, Neural ranking models with weak supervision, in: Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval, 2017, pp. 65–74, http://dx.doi.org/10.1145/3077136.3080832.

[16] S. MacAvaney, K. Hui, A. Yates, An Approach for Weakly-Supervised Deep Information Retrieval, in: SIGIR 2017 Workshop on Neural Information Retrieval, 2017.

[17] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, BERT: PRe-training of deep bidirectional transformers for language understanding, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), 2019, pp. 4171–4186, http://dx.doi.org/10.18653/v1/N19-1423.

[18] Z. Wang, P. Ng, X. Ma, R. Nallapati, B. Xiang, Multi-passage BERT: A globally normalized BERT model for open-domain question answering, 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, 2019, pp. 5878–5882, http://dx.doi.org/10.18653/v1/D19-1599.

[19] V. Karpukhin, B. Oguz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, W. t. Yih, Dense passage retrieval for open-domain question answering, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2020, pp. 6769–6781, http://dx.doi.org/10.18653/v1/2020.emnlp-main.550.

[20] R. Pradeep, R. Nogueira, J. Lin, The expando-mono-duo design pattern for text ranking with pretrained sequence-to-sequence models, 2021, arXiv:2101.05667, CoRR abs/2101.05667.

[21] A. Roberts, C. Raffel, N. Shazeer, How much knowledge can you pack into the parameters of a language model? in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP, 2020, pp. 5418–5426, http://dx.doi.org/10.18653/v1/2020.emnlp-main.437.

[22] G. Izacard, E. Grave, Leveraging passage retrieval with generative models for open domain question answering, in: Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, Association for Computational Linguistics, 2021, pp. 874–880, http://dx.doi.org/10.18653/v1/2021.eacl-main.74, Online.

[23] K. Marino, X. Chen, D. Parikh, A. Gupta, M. Rohrbach, Krisp: Integrating implicit and symbolic knowledge for open-domain knowledge-based VQA, 2020, arXiv:2012.11014, CoRR abs/2012.11014.

[24] J. Wu, J. Lu, A. Sabharwal, R. Mottaghi, Multi-modal answer validation for knowledge-based VQA, 2021, arXiv:2103.12248, CoRR abs/2103.12248.

[25] K. Marino, M. Rastegari, A. Farhadi, R. Mottaghi, OK-VQA: A Visual Question Answering Benchmark Requiring External Knowledge, in: Conference on Computer Vision and Pattern Recognition, CVPR, 2019.

[26] C.L. Clarke, E.L. Terra, Passage retrieval vs. Document retrieval for factoid question answering, in: Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval, 2003, pp. 427–428, http://dx.doi.org/10.1145/860435.860534.

[27] J. Tiedemann, J. Mur, Simple is Best: Experiments with Different Document Segmentation Strategies for Passage Retrieval, in: COLING 2008: Proceedings of the 2nd workshop on Information Retrieval for Question Answering, 2008, 17–25..

[28] W. Yang, Y. Xie, A. Lin, X. Li, L. Tan, K. Xiong, M. Li, J. Lin, End-to-End Open-Domain Question Answering with BERTserini, in: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics, Demonstrations, 2019, pp. 72–77.

[29] S. Wang, M. Yu, J. Jiang, W. Zhang, X. Guo, S. Chang, Z. Wang, T. Klinger, G. Tesauro, M. Campbell, Evidence Aggregation for Answer Re-Ranking in Open-Domain Question Answering, in: 6th International Conference on Learning Representations, ICLR 2018 - Conference Track Proceedings, 2018.

[30] P. Bhatia, L. Liu, K. Arumae, N. Pourdamghani, S. Deshpande, B. Snively, M. Mona, C. Wise, G. Price, S. Ramaswamy, X. Ma, R. Nallapati, Z. Huang, B. Xiang, T. Kass-Hout, Aws CORD-19 search: A neural search engine for COVID-19 literature, 2020, arXiv:2007.09186, CoRR abs/2007.09186.

[31] M. Bendersky, H. Zhuang, J. Ma, S. Han, K. Hall, R. McDonald, RRF102: MEeting the TREC-COVID challenge with a 100+ runs ensemble, 2020, arXiv:2010.00200, CoRR abs/2010.00200.

[32] E. Zhang, N. Gupta, R. Tang, X. Han, R. Pradeep, K. Lu, Y. Zhang, R. Nogueira, K. Cho, H. Fang, et al., Covidex: Neural ranking models and keyword search infrastructure for the COVID-19 open research dataset, 2020, arXiv:2007.07846, CoRR abs/2007.07846.

[33] P. Rajpurkar, R. Jia, P. Liang, Know what you don't know: Unanswerable questions for SQuAD, in: Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, Volume 2: Short Papers, 2018, pp. 784–789, http://dx.doi.org/10.18653/v1/P18-2124.

[34] H. Dang, J. Lin, D. Kelly, Overview of the TREC 2006 question answering track, 2008.

[35] T.R. Goodwin, D. Demner-Fushman, K. Lo, L.L. Wang, W.R. Hersh, H.T. Dang, I.M. Soboroff, Overview of the 2020 Epidemic Question Answering Track, Tech. rep., 2020.

[36] R. Tang, R. Nogueira, E. Zhang, N. Gupta, P.T.B. Cam, K. Cho, J.J. Lin, Rapidly bootstrapping a question answering dataset for COVID-19, 2020, arXiv:2004.11339, CoRR abs/2004.11339.

[37] COVID-QA: A Question Answering Dataset for COVID-19, Association for Computational Linguistics, 2020, Online.

[38] J. Borromeo, R. Pradeep, J. Lin, H2oloo At TAC 2020 : Epidemic Question Answering, Tech. rep., 2020.

[39] B. Iyer, V. Yadav, M. Franz, R.G. Reddy, A. Sultan, S. Roukos, V. Castelli, R. Florian, A. Sil, IBM Submissions To EPIC-QA Open Retrieval Question Answering on COVID-19, Tech. rep., 2020.

[40] M. Weinzierl, S.M. Harabagiu, The University of Texas At Dallas HLTRI's Participation in EPIC-QA: Searching for Entailed Questions Revealing Novel Answer Nuggets, Tech. rep., 2020.

[41] R. Johnson, T. Zhang, Effective use of word order for text categorization with convolutional neural networks, in: Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2015, pp. 103–112, http://dx.doi.org/10.3115/v1/N15-1011.

[42] Z. Yang, D. Yang, C. Dyer, X. He, A. Smola, E. Hovy, Hierarchical attention networks for document classification, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2016, pp. 1480–1489, http://dx.doi.org/10.18653/v1/N16-1174.

[43] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A.N. Gomez, Ł. Kaiser, I. Polosukhin, Attention is All You Need, in: Proceedings of the 31st International Conference on Neural Information Processing Systems, 2017, pp. 6000–6010.

[44] A. Radford, J. Wu, R. Child, D. Luan, D. Amodei, I. Sutskever, Language models are unsupervised multitask learners, 2019.

[45] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, L. Zettlemoyer, Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, in: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, 2020, pp. 7871–7880, http://dx.doi.org/10.18653/v1/2020.acl-main.703.

[46] M. Yasunaga, H. Ren, A. Bosselut, P. Liang, J. Leskovec, Qa-GNN: Reasoning with language models and knowledge graphs for question answering, in: Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, 2021, pp. 535–546, http://dx.doi.org/10.18653/v1/2021.naacl-main.45.

[47] W. Zhao, H. Peng, S. Eger, E. Cambria, M. Yang, Towards scalable and reliable capsule networks for challenging NLP applications, in: Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics, 2019, pp. 1549–1559, http://dx.doi.org/10.18653/v1/P19-1150.

[48] K. Roberts, T. Alam, S. Bedrick, D. Demner-Fushman, K. Lo, I. Soboroff, E. Voorhees, L.L. Wang, W.R. Hersh, Searching for scientific evidence in a pandemic: An overview of TREC-COVID, J. Biomed. Inform. 121 (2021) http://dx.doi.org/10.1016/j.jbi.2021.103865.

[49] J.M. Ponte, W.B. Croft, A language modeling approach to information retrieval, in: Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1998, pp. 275–281, http://dx.doi.org/10.1145/290941.291008.

[50] T. Strohman, D. Metzler, H. Turtle, W.B. Croft, Indri: A language model-based search engine for complex queries, in: Proceedings of the international conference on intelligent analysis, 2, Citeseer, 2005, 2–6..

[51] R. Krovetz, Viewing morphology as an inference process, in: Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, 1993, pp. 119–202, http://dx.doi.org/10.1145/160688.160718.

[52] I. Beltagy, K. Lo, A. Cohan, SciBERT: A pretrained language model for scientific text, in: Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, 2019, pp. 3615–3620, http://dx.doi.org/10.18653/v1/D19-1371.

[53] J. Lee, W. Yoon, S. Kim, D. Kim, S. Kim, C.H. So, J. Kang, BioBERT: a pre-trained biomedical language representation model for biomedical text mining, Bioinformatics 36 (4) (2019) 1234–1240, http://dx.doi.org/10.1093/bioinformatics/btz682.

[54] E. Alsentzer, J. Murphy, W. Boag, W.-H. Weng, D. Jin, T. Naumann, M. McDermott, Publicly available clinical BERT embeddings, in: Proceedings of the 2nd Clinical Natural Language Processing Workshop, 2019, pp. 72–78, http://dx.doi.org/10.18653/v1/W19-1909.

[55] T. Wolf, L. Debut, V. Sanh, J. Chaumond, C. Delangue, A. Moi, P. Cistac, T. Rault, R. Louf, M. Funtowicz, J. Davison, S. Shleifer, P. von Platen, C. Ma, Y. Jernite, J. Plu, C. Xu, T.L. Scao, S. Gugger, M. Drame, Q. Lhoest, A.M. Rush, Transformers: State-of-the-art natural language processing, in: Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations, Online, 2020, pp. 38–45.

[56] E. Choi, H. He, M. Iyyer, M. Yatskar, W. t. Yih, Y. Choi, P. Liang, L. Zettlemoyer, QuAC: Question answering in context, in: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, 2018, pp. 2174–2184, http://dx.doi.org/10.18653/v1/D18-1241.

[57] K. Roberts, T. Alam, S. Bedrick, D. Demner-Fushman, K. Lo, I. Soboroff, E. Voorhees, L.L. Wang, W.R. Hersh, TREC-COVID: Rationale and structure of an information retrieval shared task for COVID-19, J. Am. Med. Inf. Assoc. 27 (9) (2020) 1431–1436, http://dx.doi.org/10.1093/jamia/ocaa091.

[58] X. Saralegi, I. San Vicente, Fine-tuning BERT for COVID-19 domain ad-hoc IR by using pseudo-qrels, in: D. Hiemstra, M.-F. Moens, J. Mothe, R. Perego, M. Potthast, F. Sebastiani (Eds.), Advances in Information Retrieval, Springer International Publishing, 2021, pp. 376–383.