Double Retrieval and Ranking for Accurate Question Answering

Zeyu Zhang[1*], **Thuy Vu**[2], and **Alessandro Moschitti**[2]
[1]School of Information, The University of Arizona, Tucson, AZ, USA
[2]Amazon Alexa AI, Manhattan Beach, CA, USA
zeyuzhang@arizona.edu, {thuyvu, amosch}@amazon.com

Abstract

Recent work has shown that an answer verification step introduced in Transformer-based answer selection models can significantly improve the state of the art in Question Answering. This step is performed by aggregating the embeddings of top k answer candidates to support the verification of a target answer. Although the approach is intuitive and sound, it still shows two limitations: (i) the supporting candidates are ranked only according to the relevancy with the question and not with the answer, and (ii) the support provided by the other answer candidates is suboptimal as these are retrieved independently of the target answer. In this paper, we address both drawbacks by proposing (i) a double reranking model, which, for each target answer, selects the best support; and (ii) a second neural retrieval stage designed to encode question and answer pair as the query, which finds more specific verification information. The results on well-known datasets for Answer Sentence Selection show significant improvement over the state of the art.

1 Introduction

In recent years, automated Question Answering (QA) research has received a renewed attention thanks to the diffusion of Virtual Assistants. For example, Google Home, Siri and Alexa provide general information inquiry services, while many other systems serve customer requests in different application domains. Retrieval-based QA is enabled by two main tasks: (i) Answer Sentence Selection (AS2), which, given a question and a set of answer-sentence candidates, consists in selecting sentences (e.g., retrieved by a search engine) that correctly answer the question; and (ii) Machine Reading (MR), e.g., (Chen et al., 2017), which, given a question and a reference text, finds an exact text span that answers the question. Deploying MR systems in production is challenging for efficiency reasons, while AS2 models can efficiently target large text databases. Indeed, they originated from TREC QA tracks (Voorhees and Tice, 1999), which dealt with real-world retrieval systems since the first edition. Another limitation of MR is the focus on factoid answers: although it can in principle provide longer answers, the datasets developed for the task mainly contains short answers and in particular named entities. In contrast, as AS2 processes entire sentences, its inference steps always involve sentences/paragraphs, which make the approach agnostic to both factoid and not factoid classes.

q:	**What causes heart disease?**
c_1:	Cardiovascular disease (also called heart disease) is a class of diseases that involve the heart or blood vessels (arteries, capillaries, and veins).
c_2:	The causes of cardiovascular disease are diverse but atherosclerosis and/or hypertension are the most common.
c_3:	Cardiovascular disease refers to any disease that affects the cardiovascular system , principally cardiac disease, vascular diseases of the brain and kidney , and peripheral arterial disease.

Table 1: A question with answer candidates.

Garg et al. (2020) proposed the TANDA approach, which basically uses two stage of fine-tuning on pre-trained Transformer models (using a general dataset, ASNQ, and the target dataset), obtaining impressive improvement over the state of the art for AS2, measured on the two most used datasets, WikiQA (Yang et al., 2015) and TREC-QA (Wang et al., 2007). The approach above, based on pointwise rerankers, was significantly improved by the Answer Support-based Reranker (ASR) (Zhang et al., 2021), which adds an answer verification step similar to the one operated by fact checking systems, e.g., see the FEVER challenge (Thorne et al., 2018).

More specifically, given a question q, and a target answer, t, to be verified, which is taken from a ranked set of answer candidates $(c_1, .., c_k)$,

[*] Work done while the author was an intern at Amazon Alexa

Findings of the Association for Computational Linguistics: EACL 2023, pages 1751–1762
May 2-6, 2023 ©2023 Association for Computational Linguistics

ASR concatenates transformer-based embeddings of (q, c_i) with the max-pooling vector produced by the top k embeddings of (t, c_i), where the c_i are selected by an initial answer reranking model (e.g., TANDA). For example, Table 1 reports a question, $q =$ *What causes heart disease?*, with some candidate answers, c_1, c_2, and c_3. Selecting the correct answer c_2 is difficult, without the information: *cardiovascular disease* is also called *heart disease*. This information is provided by c_1. Thus, to compute the correctness probability of c_1, they exploit the representation of c_2, similarly to the way claims are supported in the fact verification.

ASR reduced the error of TANDA by 10% (relative), both on WikiQA and TREC-QA datasets. However, ASR shows two important limitations: first, when attempting the verification step of t, the k candidates, used in the max-pooling operation, are ranked only based on the question, i.e., independently of t. Second, the support for each t is provided by other answer candidates, which again are retrieved independently of t, i.e., t is not part of the query used for searching relevant documents.

In this paper, we provide new answer verification models, which are more efficient and accurate than ASR. We introduce a new architecture, Double Answer Reranking (DAR), which uses two models for reranking target answers and supporting candidates, respectively. Given t, the first, support reranker (SR), sorts (q, t, c_i) triplets with respect to i, in order to find the best support for t, i.e., $s_t = c_i$, while the second, answer reranker (AR), orders (q, t, s_t) triplets with respect to t, thus ranking all target answers.

Additionally, we improve the quality of supports using a second retrieval stage that searches for passages relevant to (q, t). This is important as standard answer candidates provide only information relevant to q, thus they not necessarily provide useful context for assessing t. As formulating an effective query for retrieving a question/answer pair is a new problem, and can be challenging, we exploit deep passage retrieval (DPR) (Karpukhin et al., 2020). This enables us to automatically produce embeddings for (q, t) as the target query of a neural retrieval model. As DAR is efficient, it can process many candidates from DPR, making Double Retrieval (DR) effective.

The results derived on three well-known AS2 datasets, WikiQA (Yang et al., 2015), TREC (Wang et al., 2007), and SelQA (Jurczyk et al., 2016) and a popular multi-hop QA dataset, HotpotQA (Yang et al., 2018), show consistent and significant improvement over the state of the art. For example, DAR improves TANDA by 13.6% (relative error reduction), achieving the same accuracy of the computational expensive ASR verification approach (84.36%) while DAR-DR improves the AS2 state of the art, reducing the error by an additional 8%.

We will release the datasets augmented with DPR retrieval (support candidates) for each (q, a) of each of the datasets above.

2 Related work

We focus our research on QA systems based on Information Retrieval. Since early versions, e.g., TREC QA tracks (Voorhees and Tice, 1999), these systems have been based on a search engine, which retrieves documents relevant to the asked questions, followed efficient and accurate passage rerankers to select text that most likely contains the answer. This research was revived introducing the task of answer sentence reranking (Wang et al., 2007).

In recent work, the probability, $p(q, c_i)$, for a passage/sentence, c_i, to be correct for q is estimated using neural networks, e.g., encoding q and c_i text, separately with a CNN (Severyn and Moschitti, 2015). Also designing attention mechanisms, e.g., Compare-Aggregate (Yoon et al., 2019), inter-weighted alignment networks (Shen et al., 2017). The state of the art is achieved with pre-trained Transformers, e.g., (Garg et al., 2020).

A number of researchers has proposed more than one candidate for the inference stage, e.g., using pairwise model, i.e., binary classifiers of the form $\chi(q, c_i, c_j)$, which determine the partial rank between c_i and c_j, For example, (Laskar et al., 2020; Tayyar Madabushi et al., 2018; Rao et al., 2016) use a pairwise loss and encoding. However, these methods have been largely outperformed by the pointwise models based on Transformers.

Bonadiman and Moschitti (2020) designed several joint models that improved early neural models for AS2 but failed to improve Transformer-based models. Jin et al. (2020) used the relation between candidates in Multi-task learning approach for AS2 but as they did not exploit transformer models, their results are rather lower than the state of the art. Very recently, Zhang et al. (2021) proposed ASR, a model based on a pointwise reranker fed with the embeddings refined by a pairwise approach. This significantly improved the state of the art, therefore, we analyzed ASR and specifically compare our models with it.

Very different approaches to QA systems than above use MR to extract answers from entire documents. As they have been mainly developed to find answers in a paragraph or in a text of limited size, they are rather inefficient at processing hundreds of documents, while AS2 methods can do this with high efficiency. Chen et al. (2017); Hu et al. (2019); Kratzwald and Feuerriegel (2018) proposed solutions for reliably performing inference with MR models on multiple documents. Still, the efficiency drawback was not solved. Finally, multihop QA uses multiple retrieval stages (Xiong et al., 2020; Qi et al., 2019) but the answers are just entities.

3 Baseline models for AS2

A general problem formulation for AS2 is the following: given a question q, a subset of its top-k ranked answer candidates, and a target answer $t \in C_k$, train a function, $f : Q \times C^k \rightarrow \mathbb{R}$ such that $f(q, t, c_1, .., c_{k-1})$ provides the probability of t to be correct. In this section, we describe our re-implementation of baselines, and the state-of-the-art model for AS2, namely, ASR (Zhang et al., 2021). More complex models are built on top of simpler ones, thus providing an ablation study.

3.1 Simple binary classifier (SBC)

This approach does not model dependencies between candidates, thus, we simply estimate $p(q, t)$, where $t = c_i, i = 1, \dots, k$ with a transformer-based model. Following (Garg et al., 2020), we set the input as q = Tok$_1^q$,...,Tok$_N^q$ and t =Tok$_1^t$,...,Tok$_M^t$, where we start and end the input with [CLS] and [EOS] tags, respectively, and separate sentences with [SEP]. The rest follows the standard transformer logic. We use [CLS] to represent the embedding $\mathbf{E}$ of (q, t), and we use a softmax to model the probability of the question/candidate pair classification, as $p(q, t) = softmax(W \times tanh(E(q, t)) + B)$. We fine-tune this model with log cross-entropy loss: $\mathcal{L} = -\sum_{l \in \{0,1\}} y_l \times log(\hat{y}_l)$ on pairs of text, where y_l is the correct and incorrect answer label, $\hat{y}_1 = p(q, t)$, and $\hat{y}_0 = 1 - p(q, t)$. We start training from TANDA-RoBERTa (base or large), i.e., RoBERTa fine-tuned on ASNQ (Garg et al., 2020).

3.2 Pairwise Classifier (PC)

We use the previous TANDA-RoBERTa model similarly to what is done for a multiple-choice QA (Zellers et al., 2018). We proceed as in the previous section obtaining the CLS representation for each (q, c_i) pairs. Then, for each t, we concatenate the embedding of (q, t) with all the embeddings (q, c_i), where $c_i \neq t$. This way, (q, t) is always in the first position. We train the model again using binary cross-entropy loss. At classification time, we select one candidate t at a time, set it in the first position, followed by all the others, classify all k target answers, and rerank them based on these scores.

3.3 All Candidate Multi-classifier (ACM)

We concatenate the question text with the text of all k answer candidates, i.e., $(q[SEP]c_1[SEP]c_2 \dots [SEP]c_k)$, and provide this input to the same TANDA-RoBERTa model used for SBC. We use the final hidden vector E corresponding to the first input token $[CLS]$ in a classification layer with weights $W \in R^{k \times |E|}$, and train the model using a standard cross-entropy classification loss: $y \times log(softmax(EW^T))$, where y is a one-hot vector representing labels for the k candidates, i.e., $|y| = k$. The scores for the candidate answers are calculated as $p(\{c_1, .., c_k\}) = softmax(EW^T)$. Then, we rerank c_i according their probability.

3.4 Answer Support Reranker (ASR)

The previous models have been shown to be outperformed by ASR (Zhang et al., 2021), described in Figure 1. ASR consists of five main components: (i) the primary retrieval, which recuperates documents relevant to a question and produces answer sentence candidates, (ii) an SBC, which provides the embedding of the input (q, t). This is built with the TANDA approach applied to RoBERTa pre-trained transformer (Garg et al., 2020). (iii) The joint representation of the pairs, (t, c_i), $i = 1, .., k$, $t \neq c_i$, where t and c_i are the top-candidates reranked by SBC, is obtained with a max-pooling operation over the k pairs, (t, c_i). (iv) The *Answer Support Classifier* (ASC) classifies each (t, c_i) in four classes: (0) both answer correct, (1) t is correct while c_i is not, (2) vice versa, and (3) both incorrect. This multi-classifier is trained end-to-end with the rest of the network in a multi-task learning fashion, using its specific cross-entropy loss, computed with

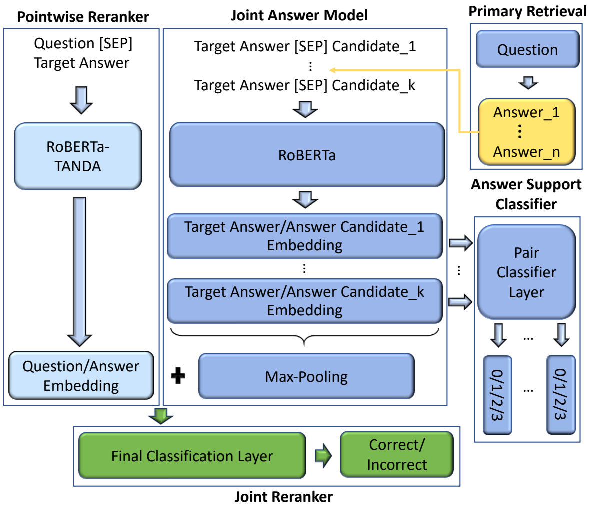
Pointwise Reranker
Question [SEP] Target Answer
RoBERTa-TANDA
Question/Answer Embedding
Joint Answer Model
Target Answer [SEP] Candidate_1
Target Answer [SEP] Candidate_k
RoBERTa
Target Answer/Answer Candidate_1 Embedding
Target Answer/Answer Candidate_k Embedding
Max-Pooling
Primary Retrieval
Question
Answer_1
Answer_n
Answer Support Classifier
Pair Classifier Layer
0/1/2/3
0/1/2/3
Final Classification Layer
Correct/ Incorrect
Joint Reranker

Figure 1: Answer Support-based Reranker (ASR)

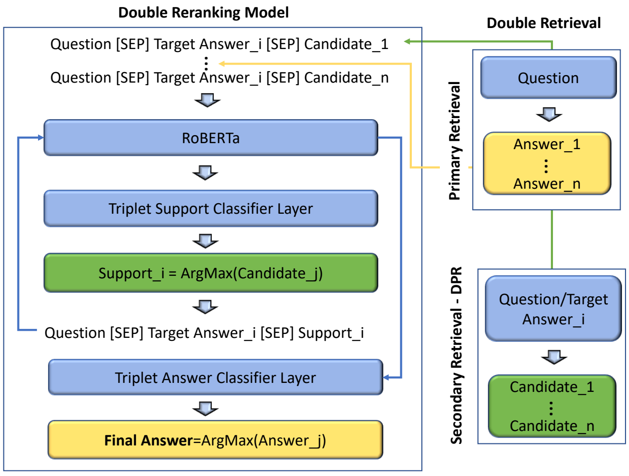
Double Reranking Model
Question [SEP] Target Answer_i [SEP] Candidate_1
Question [SEP] Target Answer_i [SEP] Candidate_n
RoBERTa
Triplet Support Classifier Layer
Support_i = ArgMax(Candidate_j)
Question [SEP] Target Answer_i [SEP] Support_i
Triplet Answer Classifier Layer
Final Answer=ArgMax(Answer_j)
Double Retrieval
Primary Retrieval
Question
Answer_1
Answer_n
Secondary Retrieval - DPR
Question/Target Answer_i
Candidate_1
Candidate_n

Figure 2: Double Answer Reranker and Retrieval (DAR-DR)

the labels above. (v) The *Final Classification Layer* takes in input the concatenation of the SBC embedding with the max-pooling embedding. Thus, the classifier scores t with respect to q, also using the other candidates.

ASC uses pre-trained RoBERTa-base (Liu et al., 2019), to generate $[CLS] \in \mathbb{R}^d$ embedding of $(q, t) = E_t$. $\hat{E}_i$ is the $[CLS]$ output of another RoBERTa-base Transformer applied to answer pairs, i.e., (t, c_i). Then, E_t is concatenated to the max-pooling tensor from $\hat{E}_1, .., \hat{E}_k$, that is, $V = [E_t : \text{Maxpool}([\hat{E}_1, .., \hat{E}_k])]$, where $V \in \mathbb{R}^{2d}$ is the final representation of the target answer t. Finally, we apply a binary classification layer: $p(y_i|q, t, c_1, .., c_{k-1}) = softmax(WV + B)$, where $W \in \mathbb{R}^{2d \times 2}$ and B are parameters to transform the representation of the target answer t from dimension $2d$ to dimension 2, which represents correct or incorrect labels.

4 Double Reranking and Retrieval

ASR is the state of the art for joint modeling candidates. However, it suffers from three main limitations: (i) it needs to limit k otherwise the complexity may be too high, this means that it may not able to process all available supporting candidates, (ii) the top k candidates are the best answer ranked by TANDA, which does not guarantee that these are also the best supports, and (iii) answer candidates may be good supports but they were not retrieved for this purpose. We address the above drawbacks proposing: (i) double reranking functions, which can efficiently rank supports as well as the best target answers, and (ii) a second stage of retrieval that performs support retrieval using a representation of the target answer and question pairs.

4.1 Double Answer Reranking (DAR)

The architecture, shown in Fig. 2, is much simpler than ASR: it just uses one RoBERTa transformer to encode triplets, question, target answer, candidate, i.e., (q, t, c_i), rather than encoding (q, t) and (t, c_i) with two separate transformer models. Then two classification layers operate two different types of ranking of the same triplets: the first, Support Ranker (SR), given t, learns to rank the best support, c_i higher. The second, Answer Ranker (AR), given the best support, i.e., $s_t = \text{arg-max}_{i:c_i \neq t}\ SR(q, t, c_i)$, learns to rank the best answer producing, $f = \text{arg-max}_{t \in C_k}\ AR(q, t, s_t)$, as the final output.

Training DAR Training SR and AR is challenging as, for the former, labels are typically not available in standard datasets. Additionally, defining a support, i.e., a piece of knowledge improving the accuracy of another classifier is not a well-understood problem. Thus, we use feedback from AR directly, i.e., a high relevant support is the one that produces the highest score in AR, if the answer is correct, and the lowest score, otherwise. We train SR and AR, at the same time, in a multi-task learning fashion, also considering that the triplets ranked by SR and AR are essentially the same: learning the different roles of SR and AR boils down from selecting a subset of triplets for their training, along with the appropriate loss function.

SR learns to rank the best supports higher. This can be enforced by requiring that s_t produces the highest score, $AR(q, t, s_t)$, among c_i scores, $\{AR(q, t, c_i)\}_i$, if t is correct, and the lowest score, otherwise. We enforce this property with a loss function: given a training example, (q, C_k), $C_k = \{c_1, .., c_k\}$, where c_i are associated with training label $l_i \in \{+1, -1\}$, we (i) select the best support, according to the current AR model, $s_t = \text{arg-max}_{i:c_i \neq t}\ l_t \times AR(q, t, c_i)$, and (ii) use the following ranking loss function to train SR:

$$L(q, c_1, \cdots, c_n) = -\log \frac{e^{\text{sim}(q,s_t)}}{\sum_{i=1}^{n} e^{\text{sim}(q,c_i)}}. \quad (1)$$

This pushes the support that provides the highest confidence score for AR in the top of the rank.

In contrast, we train AR as a standard binary classifier with the cross-entropy loss using all triplets, i.e., $(q, t, c_i) \forall t, c_i, t \neq c_i$.

4.2 Double Retrieval (DR) with DPR

The right side of Fig. 2 shows two retrieval steps: the first one is the traditional retrieval stage which, given an initial q, recuperates relevant documents, and splits them in answer sentence candidates. This step is typically carried out to build all AS2 datasets. However, if the objective is to retrieve items supporting a target t, the appropriate query should be built with the whole pair (q, t). For this reason, we propose a secondary retrieval step using (q, t). We note that (i) DAR approach does not limit the number of initial support to a fixed k as ASR does, either in training or in testing. This makes it suitable to work with more supporting items than those available from the first retrieval step. (ii) Since the semantics of (q, t) is difficult to define, neural retrieval fed with the embedding of the pair above is a promising choice.

Embeddings for support retrieval We adapted the Dense Passage Retrieval (DPR) by Karpukhin et al. (2020) for our task of support retrieval. We built two encoders $E_Q(\cdot)$ for the pairs (q, t), and $E_P(\cdot)$ for text passages p (typically they are larger than a single sentence). The encoders map the input to a d-dimensional real-valued representation, while an indexing process computes representations for all text using $E_P(\cdot)$. The retrieval of relevant content for (q, t) is done in two steps: (i) we compute the (q, t) representation using $E_Q(\cdot)$; and (ii) we then retrieve M passages that have vector representations the most similar to the pair representation, in terms of dot product:

$$\text{sim}(q, p) = E_Q(q, t)^{\intercal} E_P(p). \quad (2)$$

The encoder is trained to make the dot-product similarity corresponding to the expected ranking. Thus, for training our DPR, we use again the ranking loss in Eq. 1, where the label of p is positive if a support is part of the paragraph, i.e., $s_t \in p$.

4.3 Double Ranking and Retrieval

The combination DAR-DR needs to consider the fact that AS2 datasets do not have annotated supports. For standard datasets, we consider candidates as potential supports, where the candidates are also annotated as correct or incorrect answers. In contrast, when we retrieve new support using the (q, t) query, no label is available. However, our DAR approach does not require support labels, thus we can still train our entire DAR-DR model, by simply considering two sets: initial candidates C, on which we can train AR, and a set S containing new supports retrieved by DPR. SR can be trained on $C \cup S$, using the ranking loss (Eq. 1), which only need to estimate the best support. Again, we find it with $s_t = \text{arg-max}_i\ AR(q, t, c_i)$, where $t \in C$ and $c_i \in C \cup S \setminus t$.

5 Experiments

We compare our models with several baselines we implemented from previous work, and ASR, which is the current state of the art for AS2. For the evaluation, we used three different datasets traditionally used for AS2. Finally, we provide error analysis and model discussion.

5.1 Datasets

WikiQA is a QA dataset (Yang et al., 2015) containing a sample of questions and answer-sentence candidates from Bing query logs over Wikipedia. The answers are manually labeled. Some questions have no correct answers (*all-*), or only correct answers (*all+*). Table 2 reports the corpus statistics without $all-$ questions, and without both $all-$ and $all+$ questions (clean). We follow the most used setting: training with the $noall-$ mode and then answer candidate sentences per question in testing with the *clean* mode.

TREC-QA is another popular QA benchmark by Wang et al. (2007). Since the original test set only contain 68 questions and previous method already achieved ceiling performance (Zhang et al., 2021), we combined train., dev. and test sets, removed questions without answers, and randomly re-split into new train., dev. and test sets, which

	Train		Dev		Test	
	#Q	#A	#Q	#A	#Q	#A
no all-	873	8,672	126	1,130	243	2,351
clean	857	8,651	121	1,126	237	2,341

Table 2: WikiQA dataset statistics

contains 816, 204 and 340 questions, and 32,965, 9,591, and 13,417 question-answer pairs for the train., dev. and test sets, respectively.

SelQA is another benchmark for Selection-Based QA (Jurczyk et al., 2016), which composes about 8K factoid questions for the top-10 most prevalent topics among Wikipedia articles. We used the original splits for answer selection filed, which contain 5529 questions for train set, 785 questions for dev. set and 1590 questions for test set. SelQA is a large-scale dataset and it is more than 6 times larger than WikiQA in number of questions.

HotpotQA is a popular benchmark for multi-hop QA (Yang et al., 2018), which contains about 100,000 crowd-sourced questions that require reasoning over separate Wikipedia paragraphs. Each question not only has gold answer phrase but also has two supporting documents that contain the necessary evidence to infer the answer. To make it suitable for the AS2 task, we split paragraph into sentences, and label the sentences containing the gold answer phrase as correct answer, while considering the others as incorrect. For evaluation, we use the official dev-set-distractor as our test set.

5.2 Training and testing details

Metrics The performance of QA systems is typically measured with Accuracy in providing correct answers, i.e., the percentage of correct responses, which also refers to Precision-at-1 (P@1) in the context of reranking. We also use Mean Average Precision (MAP) and Mean Reciprocal Rank (MRR) evaluated on the test set, using the entire set of candidates for each question (this varies according to the dataset), to have a direct comparison with the state of the art.

Models We use the pre-trained RoBERTa-Base (12 layer) and RoBERTa-Large-MNLI (24 layer) models, which were released as checkpoints for use in downstream tasks[1].

Reranker training We adopt Adam optimizer (Kingma and Ba, 2014) with a learning rate of 2e-5 for the transfer step on the ASNQ dataset (Garg et al., 2020), and a learning rate of 1e-6 for the adapt step on the target dataset. We apply early stopping on the development set of the target corpus for both fine-tuning steps based on the highest MAP score. We set the max number of epochs equal to 3 and 9 for the adapt and transfer steps, respectively. We set the maximum sequence length for RoBERTa to 128 tokens.

ASR training Again, we use the Adam optimizer with a learning rate of 2e-6 for training the ASR model on the target dataset. We utilize one Tesla V100 GPU with 32GB memory and a train batch size of eight. We use two transformer models for ASR: a RoBERTa Base/Large for PR, and one for the joint model (see Fig. 1). We set the maximum sequence length for RoBERTa to 128 tokens and the number of epochs as 20. We select the best k chosen in (Zhang et al., 2021).

DAR implementation and training For training the DAR model, we also use the Adam optimizer but with a different learning rate, 5e-6. We utilize two Tesla A100 GPUs with 40GB memory and a train batch size of 128. DAR only needs one transformer model: a RoBERTa Base/Large (see Fig. 2). The maximum sequence length and the number of epochs are the same with ASR training, which are 128 and 20 separately.

DPR implementation and training We utilize the same training configuration of the original DPR in Karpukhin et al. (2020). Then, we used it to build a large index having up to 130MM passages extracted from 54MM documents of CommonCrawl[2]. We selected English Web documents of 5,000 most popular domains, including Wikipedia, from the recent releases of Common Crawl of 2019 and 2020. We then filtered pages that are too short or without proper HTML structures, i.e., having title and content. To retrieve to N candidates, we input our DPR with (q, t) pairs as query to retrieve top 1000 passages.

DAR-DR implementation and training The training configuration is similar to DAR training with the different steps highlighted in Sec. 4.2. For each (q, c_i) of our datasets, we used our DPR for retrieving 1000 supporting paragraphs, which are then split into sentences, s. We rank s according to a $E_Q(q,t) \cdot E_P(s)$, where $E_P(s)$ provides the

[1] https://github.com/pytorch/fairseq

[2] commoncrawl.org

RoBERTa Base	WikiQA				TREC-QA				SelQA				HotpotQA			
	P@1	RER	MAP	MRR	P@1	RER	MAP	MRR	P@1	RER	MAP	MRR	P@1	RER	MAP	MRR
TANDA (Garg et al.)	–	–	0.8890	0.9010	–	–	–	–	–	–	–	–	–	–	–	–
ASR (Zhag et al.)	0.8436	13.64%	0.9014	0.9123	–	–	–	–	–	–	–	–	–	–	–	–
SBC	0.8189	0.00%	0.8860	0.8983	0.8824	0.00%	0.8979	0.9277	0.9302	0.00%	0.9512	0.9587	0.6598	0.00%	0.7576	0.7685
ACM	0.7819	-20.43%	0.8542	0.8684	0.8824	0.00%	0.8942	0.9272	0.9308	0.86%	0.9511	0.9589	0.6597	-0.03%	0.7574	0.7681
PC	0.8272	4.58%	0.8927	0.9045	0.8882	4.93%	0.9000	0.9319	0.9302	0.00%	0.9514	0.9587	0.6718	3.53%	0.7644	0.7750
ASR (ours)	0.8436	13.64%	0.9014	0.9123	0.9088	22.45%	0.9036	0.9420	0.9314	1.72%	0.9519	0.9592	0.6795	5.79%	0.7724	0.7812
ASR-Rank	0.8436	13.64%	0.9012	0.9108	0.9088	22.45%	0.9181	0.9445	0.9296	-0.86%	0.9503	0.9580	0.6768	5.00%	0.7742	0.7824
DAR	0.8519	18.22%	0.9011	0.9136	0.9118	25.00%	0.9181	0.9446	0.9415	16.19%	0.9592	0.9653	**0.6844**	**7.23%**	**0.7754**	**0.7854**
DAR-DR	**0.8560**	**20.49%**	**0.9051**	**0.9164**	**0.9176**	**29.93%**	**0.9233**	**0.9493**	**0.9484**	**26.07%**	**0.9616**	**0.9687**	0.6832	6.88%	0.7729	0.7832

Table 3: Performance of different models using RoBERTa base Transformer on WikiQA, TRECQA, SelQA and HotpotQA. RER is the relative error reduction on P@1. The difference between P@1 of DAR and DAR-DR and P@1 of all the other systems is statistically significant at 95%.

embedding representation of each s, even though we trained $E_P(\cdot)$ for passages. We select the top 10 sentences as support for all the experiments with DAR-DR. It should be noted that all datasets for retrieval-based QA are based on candidates retrieved with an initial search engine, e.g., Bing, Google, TREC systems. This constitutes the first standard retrieval in our DR approach.

5.3 Comparative/ablated results

We design a set of baselines (see Sec. 3), which also constitute the best ablation systems of our most complex architecture DAR-DR. Indeed, **SBC** is our reimplementation of TANDA, which corresponds to the basic system (or basic component) of our architecture, it uses only one reranker and no joint inference. **PC** is the simplest joint model, which still uses only one classifier as SBC but applied to pairs of answers. **ASR** (ours) is our reimplementation of ASR, which uses an SBC model, a PC model, and an internal SR (called ASC) model as in DAR, used just for classification, no ranking. **ASR-Rank** extends ASR using the top 3 candidates re-ranked by ASC category 0 score (see Sec. 3.4), instead of using the standard TANDA rank. We introduced, ASR-Rank to show an approach similar to DAR. **ACM** is a joint model over all k candidates (theoretically more expressive than just joint models over pairs). **DAR** uses two rerankers as ASR-Rank but only one transformer and our approach to train them. Finally, **DAR-DR** adds to DAR new candidates retrieved by DPR.

Main results Table 3 reports P@1, MAP and MRR of models on WikiQA, TREC-QA, SelQA and HotpotQA datasets. TANDA and ASR rows report the results obtained by Garg et al. (2020) and Zhang et al. (2021), respectively, which certify the alignment between our and previous work setting and implementation. We note that:

(i) P@1, MAP and MRR correlate well, thus, we can focus our analysis on P@1, which typically provides the QA performance. The AS2 model P@1 numbers are in the lower 80s% for all datasets but HotpotQA. This means that absolute improvements are not expected to be large, thus we also report the relative error reduction (RER) for P@1, which better shows model differences.

(ii) Our SBC and ASR replicate the performance reported in previous work (WikiQA and TREC-QA), which are the previous state of the art.

(iii) We confirm that ASR, using candidate pairwise information greatly improves on single answer classification models, e.g., we observe a relative error reduction of 13.64% (from 81.89 to 84.36) over TANDA and SBC, which do not use the information from other candidates.

(iv) Our proposed model DAR significantly reduces the error of QA systems with respect to ASR by 4.58% (from 84.36 to 85.19), 2.55% (from 90.88 to 91.18), 14.47% (from 93.14 to 94.15), and 1.44% (from 67.95 to 68.44) on WikiQA, TREC-QA, SelQA, and HotpotQA, respectively. It is interesting to note that DAR only uses the half of the parameters of ASR (125M vs. 250M). The combination between the two rerankers for answer and support generates more selective information than max-pooling pairwise embeddings.

(v) To verify that the unique feature of DAR of effectively combining training examples and their losses is a key element, we implemented ASR-Rank, which also selects supporting candidates for ASR, using its internal answer pair classifier, ASC(t, c_i). The results derived on WikiQA and TREC-QA show no difference between ASR and ASR-Rank, while the latter underperforms on SelQA. This shows that the improvement produced by DAR is not about selecting the best support in absolute, but it is about selecting the support that

Roberta Large	WikiQA				
	P@1	RER	MAP	MRR	Param.
SBC	0.8724	0.00%	0.9151	0.9266	355M
ASR	0.8971	19.36%	0.9280	0.9399	710M
DAR	0.8889	12.93%	0.9230	0.9362	355M
DAR-DR	0.8930	16.14%	0.9241	0.9375	355M

Table 4: Results on WikiQA using RoBERTa Large.

can produce the highest confidence in the answer selector (see Sec. 4.1).

(vi) DAR-DR introduces 10 additional supports to DAR processing, retrieved with our modified DPR approach. These new candidates do not have any label indicating if they are good or bad support. They are automatically ranked with the DAR approach. The results show an RAR of 2.27%, 4.93%, and 9.88%, on WikiQA, TREC-QA, and SelQA, respectively. Suggesting that retrieving supporting candidates for (q, t) can be very effective. HotpotQA does not benefit from retrieving candidates external to the dataset as the original candidate set always contains at least one correct support by construction, thus no additional retrieval is needed.

(vii) Finally, we perform randomization test (Yeh, 2000) to verify if the models significantly differ in terms of prediction outcome. Specifically, for each model, we compute the best answer for each question and derive binary output based on the ground truth. We then follow the randomization test to measure the statistical significance between two models. We use 100,000 trials for each calculation. The test show statistical significant difference of DAR and DAR-DR vs. all the other models over all datasets but HotpotQA, with p < 0.05, and between DAR and DAR-DR on SelQA.

Results with large models We experimented with SBC, ASR, DAR and DAR-DR models implemented on a larger transformer, i.e., RoBERTa Large, on WikiQA. Table 4 reports the comparative results: SBC and ASR replicate the results by Zhang et al. (2021), i.e., a P@1 of 87.24% and 89.71%, respectively; the latter is the state of the art on WikiQA with a P@1 of 89.71%. Both DAR and DAR-DR improve SBC up to 20% RAR. However, even DAR-DR is behind ASR, by about 3.21% of RER. This different outcome with respect previous results on the RoBERTa base can be explained by looking at the column reporting model parameters. As before, ASR uses the double of parameters of DAR, however, in this case the number of parameters is 710M, which is a large number in absolute: although DAR is a better model, it can hardly improve a model with 355M parameters more.

q:	**what is the measurements of saturn 's moons?**
c_1:	The rings of Saturn are made up of objects ranging in size from microscopic to hundreds of meters, each of which is on its own orbit about the planet.
c_2:	Saturn has 62 moons with confirmed orbits , 53 of which have names and only 13 of which have diameters larger than 50 kilometers.
c_3:	The moons of Saturn (also known as the natural satellites of Saturn) are numerous and diverse ranging from tiny moonlets less than 1 kilometer across to the enormous Titan which is larger than the planet Mercury.
c_4:	Saturn has seven moons that are large enough to be ellipsoidal due to having planetary mass , as well as dense rings with complex orbital motions of their own.

Table 5: A question with answer candidates; c_2 and c_3 are correct.

5.4 Model discussion and error analysis

Tab. 5 shows a question with the rank provided by SBC. The top-1 answer, c_1 is incorrect, as it refers to objects of Saturn's rings, instead of targeting its moons. SBC probably got tricked by the phrase *ranging in size*. ASR also selected c_1 using the support of the top 3 candidates selected by SBC, i.e., c_2, c_3, and c_4. These candidates support c_1 as they provide more context, e.g., *moon*, which is not in c_1 but it is required in the question. The main problem of ASR is the fact that correct answers also tend to support imperfect but reasonable answers such as c_1. In contrast, for each t, DAR learns to select the best support: in the example, it selects the correct answer c_2 using c_4 as support. This probably provides phrases such as *seven moons that are large enough* supporting c_2 phrases such as *have diameters larger than*.

In Tab. 6, we see an example, in which SBC ranks an incorrect answer at the top. It probably prefers c_1 to the correct answer c_2 because it matches the main question entity and verb, i.e., *Family Guy* and *premier*, while c_2 does not contain explicit reference to the main entity. Also ASR and DAR cannot select c_2, as the available supports, c_1 and c_3, do not provide any useful information. In contrast, DAR-DR can use new retrieved support, i.e., s_1, which contains the main entity and reinforces the information in c_2, i.e., *22 millions*.

See Appendix for more discussion.

6 Conclusion

In this paper, we propose, DAR, a transformer architecture based on two reranking heads: (i) the answer reranker (AS2 model) and the answer support

q:	**How many viewers did "Family Guy" premier to?**
c_1:	Family Guy officially premiered after Fox's broadcast of Super Bowl XXXIII on January 31, 1999, with "Death Has a Shadow.
c_2:	The show debuted to 22 million viewers, and immediately generated controversy regarding its adult content.
c_3:	At the end of its first season, the show was #33 in the Nielsen ratings, with 12.8 million households tuning i.
s_1:	Family Guy has been around since 1999 with 11 seasons to date, the viewing rates have dropped from over 22 millions to 7 million.

Table 6: Example with c_2 correct.

reranker. We optimize the latter imposing a loss function that penalizes non optimal support for the target answer, thus avoiding the need of defining and manually labeling supporting data. Additionally, we introduce a second retrieval stage based on DPR, where we optimize the score function between answer/question pair and the retrieving passage. The experiments with four well-known datasets show consistent improvement of DAR over the state of the art, and the potential benefit of the secondary retrieval, achieving up to 14.47 of relative error reduction (on SelQA). We will release software, models, and the DPR retrieved data for all datasets for fostering research in this field.

Limitations

We propose a new QA architecture that operate a second retrieval. This can make the approach slower than a standard QA system using only one retrieval but, at the same time, it enables the possibility to retrieve critical information. The latter can be used to verify question/answer pairs or also complement the information need of the user. This is clearly a future direction for QA/personal assistant systems. As we explain in the paper, we designed a DPR model which can specifically retrieve supporting items (no just answer candidates), as we can query DPR with the pair (question, answer to be verified). This is a major novelty with respect to systems that can only retrieve text relevant to the question.

Our new approach uses only one support to verify answer correctness. This may be seen as lack of exploration of the model potential. However, using one support only requires a classifier of the form SR(q, t, s_i). If we use more supports, for example two, we will have a classifier of the type SR(q, t, s_i, s_j). This means that to find the arg-max we would need to iterate over k^2, where k is the number of candidates (so in general k^n with n the number of supports we want to use). This is much less efficient than our approach. Although, approximated solutions more efficient than O(k^n) can be surely designed, in this paper, we have focused on a rather efficient version, which has also shown to improve the state of the art.

References

Daniele Bonadiman and Alessandro Moschitti. 2020. A study on efficiency, accuracy and document structure for answer sentence selection. *CoRR*, abs/2003.02349.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *CoRR*, abs/1704.00051.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2020. TANDA: transfer and adapt pre-trained transformer models for answer sentence selection. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*, pages 7780–7788. AAAI Press.

Minghao Hu, Yuxing Peng, Zhen Huang, and Dongsheng Li. 2019. Retrieve, read, rerank: Towards end-to-end multi-document reading comprehension. *CoRR*, abs/1906.04618.

Zan-Xia Jin, Bo-Wen Zhang, Fang Zhou, Jingyan Qin, and Xu-Cheng Yin. 2020. Ranking via partial ordering for answer selection. *Information Sciences*.

T. Jurczyk, M. Zhai, and J. D. Choi. 2016. Selqa: A new benchmark for selection-based question answering. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 820–827.

Vladimir Karpukhin, Barlas Oguz, Sewon Min, Patrick S. H. Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*, pages 6769–6781. Association for Computational Linguistics.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980.

Bernhard Kratzwald and Stefan Feuerriegel. 2018. Adaptive document retrieval for deep question answering. In *EMNLP'18*, pages 576–581.

Md Tahmid Rahman Laskar, Jimmy Xiangji Huang, and Enamul Hoque. 2020. Contextualized embeddings based transformer encoder for sentence similarity modeling in answer selection task. In *Proceedings of*

The 12th Language Resources and Evaluation Conference, pages 5505–5514, Marseille, France. European Language Resources Association.

Yinhan Liu, Myle Ott, Naman Goyal, Jingfei Du, Mandar Joshi, Danqi Chen, Omer Levy, Mike Lewis, Luke Zettlemoyer, and Veselin Stoyanov. 2019. Roberta: A robustly optimized BERT pretraining approach. *CoRR*, abs/1907.11692.

Peng Qi, Xiaowen Lin, Leo Mehr, Zijian Wang, and Christopher D. Manning. 2019. Answering complex open-domain questions through iterative query generation. 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)*, pages 2590–2602, Hong Kong, China. Association for Computational Linguistics.

Jinfeng Rao, Hua He, and Jimmy Lin. 2016. Noise-contrastive estimation for answer selection with deep neural networks. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1913–1916.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *SIGIR'15*.

Gehui Shen, Yunlun Yang, and Zhi-Hong Deng. 2017. Inter-weighted alignment network for sentence pair modeling. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1179–1189, Copenhagen, Denmark. Association for Computational Linguistics.

Harish Tayyar Madabushi, Mark Lee, and John Barnden. 2018. Integrating question classification and deep learning for improved answer selection. In *Proceedings of the 27th International Conference on Computational Linguistics*, pages 3283–3294, Santa Fe, New Mexico, USA. Association for Computational Linguistics.

James Thorne, Andreas Vlachos, Christos Christodoulopoulos, and Arpit Mittal. 2018. FEVER: a large-scale dataset for fact extraction and VERification. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long Papers)*, pages 809–819, New Orleans, Louisiana. Association for Computational Linguistics.

E. Voorhees and D. Tice. 1999. *The TREC-8 Question Answering Track Evaluation*, pages 77–82. Department of Commerce, National Institute of Standards and Technology.

Mengqiu Wang, Noah A. Smith, and Teruko Mitamura. 2007. What is the Jeopardy model? a quasi-synchronous grammar for QA. In *EMNLP-CoNLL'07*, pages 22–32, Prague, Czech Republic. Association for Computational Linguistics.

Wenhan Xiong, Xiang Li, Srini Iyer, Jingfei Du, Patrick Lewis, William Yang Wang, Yashar Mehdad, Scott Yih, Sebastian Riedel, Douwe Kiela, et al. 2020. Answering complex open-domain questions with multi-hop dense retrieval. In *International Conference on Learning Representations*.

Yi Yang, Wen-tau Yih, and Christopher Meek. 2015. Wikiqa: A challenge dataset for open-domain question answering. In *Proceedings of the 2015 conference on empirical methods in natural language processing*, pages 2013–2018.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. 2018. HotpotQA: A dataset for diverse, explainable multi-hop question answering. In *Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Alexander S. Yeh. 2000. More accurate tests for the statistical significance of result differences. *CoRR*, cs.CL/0008005.

Seunghyun Yoon, Franck Dernoncourt, Doo Soon Kim, Trung Bui, and Kyomin Jung. 2019. A compare-aggregate model with latent clustering for answer selection. *CoRR*, abs/1905.12897.

Rowan Zellers, Yonatan Bisk, Roy Schwartz, and Yejin Choi. 2018. Swag: A large-scale adversarial dataset for grounded commonsense inference. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP)*.

Zeyu Zhang, Thuy Vu, and Alessandro Moschitti. 2021. Joint models for answer verification in question answering systems. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing, ACL/IJCNLP 2021, (Volume 1: Long Papers), Virtual Event, August 1-6, 2021*, pages 3252–3262. Association for Computational Linguistics.

A Deeper Discussion

A.1 Double Retrieval

The complete architecture DAR-DR can operate a second retrieval, which can make the approach slower than a standard system using only one retrieval but, at the same time, it enables the possibility to retrieve critical information. The latter can be used to verify question/answer pairs or also complement the information need of the user. This is clearly a future direction for QA/personal assistant systems. As we explain in the paper, we designed a DPR model which can specifically retrieve supporting items (no just answer candidates), as we can query DPR with the pair (question, answer to be verified). This is a major novelty with respect to systems that can only retrieve text relevant to the question.

It should be noted that we apply two answer sentence retrieval steps: (i) the standard one, which is contained in all datasets for AS2 based QA systems. See our description of WikiQA, TREC, SelQA, and HotpotQA. For example, WikiQA uses Bing to retrieve passages. (ii) Our innovative retrieval based on our new DPR model. This takes (q, a) as query and returns passages that have higher probability to be good support for a with respect to q.

Our DAR-DR aims to be an end-to-end system, AS2 tasks are defined using retrieval systems. We also operate the second retrieval. In other words, a DAR-DR system deployed in production will always performs 2 stages of retrieval to provide answers to users.

A.2 AS2 Tradition

Please note that the AS2 research our paper builds on has been contributed for more than 20 years. It started in TREC competitions (QA track 1999). It has been revived in 2007 with the specialization of passage reranking in answer sentence selection (AS2): see for example the systems based on TREC data https://aclweb.org/aclwiki/Question_Answering_(State_of_the_art). Also note that TANDA by Garg et al, 2020 is simply a transformer fined-tuned in two steps (i) on ASNQ dataset proposed by the same authors, and (ii) on the target dataset.

A.3 Ablation study

The baseline models we implemented and compared to are ablated versions of our systems, sometimes including different alternatives (instead of just excluding some features). Sec. 5.3 explains how the different models we test constitute an excellent ablation study.

A.4 Usefulness of reporting result with Relative Error Reduction

The relative error reduction is suitable for reporting the performance in our setting since we are improving state-of-the-art systems with performance ranging from ∼81% to ∼97% (depending on the measure and datasets). Reporting absolute (or also relative) improvement does not capture the complexity of the task. For example, improving a system from 30% to 31% (margin of improvement 70%) is completely different than improving a system from 98% to 99%, where the margin of improvement is 2%. Relative error reduction, which we use, accounts for such difficulties. In any case, whatever lens one uses, the results are statistical significant, showing that we improve the state of the art.

A.5 Model Effectiveness

We report the number of model parameters on the Table 4, which shows that our solution uses half of the parameters of previous state of the art, ASR (indeed that uses two transformer models: instead of our DAR only uses one).

A.6 Multiple supports

Using one support only requires a classifier of the form SR(q, t, s_i). If we use more supports, for example two, we will have a classifier of the type SR(q, t, s_i, s_j). This means that to find the arg-max we would need to iterate over k^2, where k is the number of candidates (so in general k^n with n the number of supports we want to use). This is much less efficient than our approach. Although, approximated solutions more efficient than O(k^n) can be surely designed, in this paper, we have focused on a rather efficient version, which has also shown to improve the state of the art.

Moreover, although it may happen that to verify information multiple pieces are required, this situation is rather rare in general open QA, as the web contains the needed information in a redundant fashion. This means we can most times retrieve a compact version of an answer *why should it be available only in a fragmented way?*

For other more specific application scenarios, e.g., deriving answers from several axioms and logic formulas (expressed in text format), combinations of different supports, composing different retrieved pieces may be required. However, this scenario is out of the scope of our paper: it can be an interesting new research.

A.7 Comparison with (Zhang et al., 2021)

(Zhang et al., 2021) is a great work which outperforms the previous state of the art in AS2, i.e., TANDA, which seemed very difficult to improve. Our contributions:

- We defined a new techniques to automatically learn to rank support without using any annotation, which is for example used in hotpotQA.

- Our approach outperforms (Zhang et al., 2021) on base architectures. With respect to LARGE, we did not have the language models comparing with the same number of parameters but using 355M parameters less, our approach provide close results.

- Our RoBERTa-base model only use half of the parameters of (Zhang et al., 2021), which these days of energy crisis is absolutely important results.

- Most importantly, we defined a new paradigm for QA (and answer verification), which uses double retrieval, our support reranker can be used to select support obtained with a second stage of retrieval. This to our knowledge is completely new for answer sentence selection.

Our approach improves all previous techniques in a fair comparison, which means similar number of parameters. In case of RoBERTa based, our approach outperforms models with the double of parameters. Specifically, only using 130M parameters, it outperforms an architecture of 260M parameters, i.e., architectures having 130M parameters more. The fact that our approach does not perform an architecture of 710M parameters, i.e., a model that used 355M more than ours, is not a limitation. We show these unfair experiments (for our approach) because they provide strong evidence about the effectiveness of our approach. The lower performance on HotpotQA is expected as the related task is not answer sentence selection, for which our approach was built. Indeed, HotpotQA focused on entities and annotated data such that two paragraphs complement each other, which is a more restrictive assumption than our approach. For the sake of generality, we showed that our approach can also work well for this rather different setting.