A Study on Efficiency, Accuracy and Document Structure for Answer Sentence Selection

Daniele Bonadiman*
Amazon AI
dbonadim@amazon.com

Alessandro Moschitti
Amazon Alexa
amosch@amazon.com

Abstract

An essential task of most Question Answering (QA) systems is to re-rank the set of answer candidates, i.e., Answer Sentence Selection (AS2). These candidates are typically sentences either extracted from one or more documents preserving their natural order or retrieved by a search engine. Most state-of-the-art approaches to the task use huge neural models, such as BERT, or complex attentive architectures. In this paper, we argue that by exploiting the intrinsic structure of the original rank together with an effective word-relatedness encoder, we achieve the highest accuracy among the cost-efficient models, with two orders of magnitude fewer parameters than the current state of the art. Our model takes 9.5 seconds to train on the WikiQA dataset, i.e., very fast in comparison with the $\sim$ 18 minutes required by a standard BERT-base fine-tuning.

1 Introduction

In recent years, there has been a renewed interest in Question Answering (QA) led by industrial needs, e.g., the development of personal assistants and academic research on neural networks. Regarding the latter, AS2 (Wang et al., 2007; Yang et al., 2015) and Machine Reading Comprehension (MRC) (Richardson et al., 2013; Rajpurkar et al., 2016) have been largely explored.

AS2 consists of selecting sentences that are answers to a target question from documents or paragraphs retrieved from the web by a search engine. MRC regards the extraction of an exact text span from a document answering the question, where the document is usually provided with the target question.

Even though MRC is gaining more and more popularity, AS2 is more suitable for a production scenario, where a combination of a retrieval engine and an automatic sentence selector can already constitute a QA system. In contrast, MRC has been mainly developed to find answers in a paragraph or a text of limited size. Several models for adapting MRC to an end-to-end retrieval setting have been proposed, e.g., Chen et al. (2017a) and Kratzwald and Feuerriegel (2018), the deployment of MRC systems in production is challenged by two key factors: the lack of datasets for training MRC with realistic retrieval data, and the large volume of content needed to be processed, i.e., MRC cannot efficiently process a large amount of retrieved data.

In contrast, AS2 research originated from the TREC competitions (Wang et al., 2007); thus, it has targeted large databases of unstructured text from the beginning of its development. Neural models have significantly contributed to AS2 with new techniques, e.g., Wang and Jiang (2016), Qiao et al. (2019) and Nogueira and Cho (2019). Recently, new approaches for pre-training neural language models on large amount of data, e.g., ELMO (Peters et al., 2018), GPT (Radford et al.,), BERT (Devlin et al., 2019), RoBERTa (Liu et al., 2019), have led to major advancements in several NLP subfields. These pre-training techniques allow for creating models that automatically capture dependencies between the sentence compounds. Interestingly, the resulting models can be easily adapted to different tasks by fine-tuning them on the target training data.

Unfortunately, all the models above (especially Transformer-based architectures) require a considerable number of parameters (up to 340 million for BERT Large). This poses three critical challenges for having

*work done at University of Trento prior to joining Amazon

Proceedings of the 28th International Conference on Computational Linguistics, pages 5211–5222
Barcelona, Spain (Online), December 8-13, 2020

How long was I Love Lucy on the air ?
I Love Lucy is an American television sitcom starring Lucille Ball , Desi Arnaz , Vivian Vance , and William Frawley .
The black-and-white series originally ran from October 15, 1951, to May 6, 1957, on the Columbia Broadcasting System (CBS).
After the series ended in 1957, however, a modified version continued for three more seasons with 13 one-hour specials, running from 1957 to 1960, known first as The Lucille Ball-Desi Arnaz Show and later in reruns as The Lucy–Desi Comedy Hour .
I Love Lucy was the most watched show in the United States in four of its six seasons, and was the first to end its run at the top of the Nielsen ratings (an accomplishment later matched by The Andy Griffith Show and Seinfeld).
I Love Lucy is still syndicated in dozens of languages across the world

Table 1: An example of question/answer-candidate from WikiQA. In green the answer to the question.

such models in production: firstly, it requires powerful GPUs to achieve an acceptable service latency; secondly, although the classification of candidates can be parallelized, the necessary number of GPUs will prohibitively increase the operational cost and creating substantial environmental issues, as pointed out in Strubell et al. (2019); last, transformer-based architectures require many resources for pre-training, e.g., both data and compute power (TPUs). These resources may not be available for low resource languages or domain-specific applications.

In this paper, we study and propose solutions to design accurate AS2 models, still preserving high efficiency. We first note that (i) the primary source of inefficiency is, unfortunately, the contextual embedding, e.g., language models produced by Transformer networks or other methods such as ELMo. These introduce at least one order of magnitude more parameters in the AS2 models than standard models based on CNNs or LSTMs. (ii) The other significant source of inefficiency is the attention mechanism. As both of the above features critically impact accuracy, we provide an alternative approach to preserve it as much as possible. In particular, we jointly model all candidates for a given question to capture the global structure of the document or the rank as provided in input to the model. Our experiments verify the hypothesis that in several AS2 datasets, the data often presents an underlying ranking structure that refers not only to the relations between a question and all its candidates but also to the inter-dependencies among the candidates themselves. Additionally, the availability of the representation of several candidates can help weaker models to improve their inference, e.g., in case there are two correct and similar answers, they can get strength from each other, or even a wrong answer sentence can provide additional context to make a better inference.

Our approach can both (i) capture structure in the original rank and (ii) exploit the representation of multiple candidates. For this purpose, we show that it is essential to implement two main logic blocks: (i) an encoder able to capture the relation between the question and each of its candidates, e.g., using an attention mechanism; and (ii) the structure of the sentences in the original rank, which is the original order of the sentences in the document or in the rank retrieved by the search engine. Regarding the second block, we use an additional layer constituted by a bidirectional recurrent neural network (BiRNN), which is fed with the representation of question/answer candidate pairs. The latter are joint representations of the question and the answer obtained by the question-answer encoder. For the attention mechanism, we propose *Cosinet*, a network that uses a sort of *static attention*, given by the cosine similarity between the embedding representation of the question and answer words. We show that this solution is very efficient and does not cause almost any drop compared to the use of standard attention.

We tested our models on four different datasets, the well-known WikiQA dataset, the adaptation of SQuAD (Rajpurkar et al., 2016), and Natural Questions (Kwiatkowski et al., 2019) datasets to the AS2 task. Our comparative experiments show that (i) our models achieve better results than other efficient approaches, but (ii) the lack of contextual embeddings prevents the model from learning more complex functions, leading to lower results than the very expensive solutions. We partially solve this problem by using our joint model, which significantly improves the accuracy of efficient methods. Despite this adds a small overhead during training and testing, it is crucial to capture the structure of data. For example, the results on WikiQA show that BiRNN added to the Cosinet improves of ~ 4 points previous baselines.

Finally, our word-relatedness encoder can replace the standard attention to enhance the speed of the fast attention-based approaches, resulting in a fast and accurate network in the class of fast methods. Our research will gain more and more important also in the light of improving the efficiency of large architecture using our models for sequential re-ranking (Matsubara et al., 2020; Soldaini and Moschitti, 2020).

2 AS2: Answer Sentence Selection

The task of Answer Sentence Selection (AS2) can be formalized as follows: given a question q and a set of answer sentence candidates $C = \{c_1, c_2, ..., c_n\}$, assign a score s_i for each candidate c_i such that the sentence receiving the highest score is the one that most likely contains the answer, i.e., the answer sentence. It is interesting to note that AS2 can be, in fact, modeled as a re-ranking task.

Although re-ranking is a structured output problem, most state-of-the-art approaches treat it as point-wise classification, i.e., classifying answer sentences as positive and all the others as negative. This design bias prevents AS2 models from capturing the underlying structure of the original rank. However, in this paper, we argue that building systems capable of capturing such information is crucial for improving the performance of efficient AS2 models.

2.1 AS2 Datasets

AS2 datasets can be divided into two categories: retrieval based and document-based. The difference between the two categories resides in the source of the answer candidates. In the former, answer candidates are retrieved from a search engine, i.e., TrecQA (Wang et al., 2007) and, more recently, MSMarco (Bajaj et al., 2016). For the latter, a search engine is often used to retrieve relevant documents, but the task is to select the relevant answer candidate from the document itself. Notable examples of document-based AS2 are the WikiQA dataset (Yang et al., 2015) and the "long-answer" version of Natural Question (Kwiatkowski et al., 2019). Although the datasets are different in terms of annotation, they seem to require similar features for detecting relevant answers in the candidate set. These are essentially the lexical overlap between the question and the answer candidate as well as the global candidate structure, i.e., the original order of the sentences in the rank.

2.1.1 Lexical Overlap

One of the most reliable features in AS2 datasets is the lexical overlap, i.e., whether words appear in both questions and answer candidates.

The importance of this feature is highlighted in Table 2. We used the number of unique words in both the question and the candidates as a single feature, which is called word-overlap WO, to rank question-answer pairs. From the table, it is clear that this feature alone significantly outperforms the random baseline in most datasets. For SQuAD-sent, our adaptation of the SQuAD v. 2.0 dataset to the task of identifying the sentence containing the answer, this feature alone identifies the answer sentence 65.48% of the times. All recent work models have tried to model such features; for example, Severyn and Moschitti (2016) uses a relational feature that marks words appearing in both the question and the answer, and many state-of-the-art approaches have primarily used the attention mechanism (Wang and Jiang, 2016; Bian et al., 2017; Sha et al., 2018).

	WikiQA	SQuAD-sent	NQ-LA
# questions (Q)	633	11873	6230
# sentences (C)	6165	63959	193k
% Q answered	38.39	49.92	55.47
avg. # passages	9.74	5.38	30.95
avg. Q length	7.28	10.02	9.38
avg. C length	25.36	23.75	98.76
P@1 (random)	14.43	18.34	3.24
MAP (random)	25.15	43.81	12.33
P@1 (WO)	32.51	65.48	23.06
MAP (WO)	51.02	77.90	38.08
P@1 (RR)	46.09	30.54	46.06
MAP (RR)	64.21	53.53	57.30
P@1 (WO+RR)	56.38	73.12	41.01
MAP (WO+RR)	68.25	83.60	53.98

Table 2: Statistics of the different datasets (the test-set are taken into account).

2.1.2 Global Structure

Another relevant feature for the AS2 datasets is the global structure present in the original rank. The structure of the document provides an essential signal for AS2. Table 2 shows that, in the case of WikiQA, SQuAD, and Natural Questions, there is a high chance that the answer is contained in the first sentence/paragraph. This is particularly true for WikiQA and Natural Questions. In these datasets, the Precision at one (P@1), computed on the original sentence rank (RR) (order of the sentences in the raw text), is ~ 46. There may be several reasons for this distribution. For example, we believe that there is an intrinsic correlation between the real-world distribution of questions and the structure of the Wikipedia document: encyclopedic knowledge is usually organized such that more general information about a topic is summarized and organized at the beginning of the document.

In contrast, the signal is less present in other datasets, such as SQuAD, where annotators are asked to write questions after reading the whole paragraph, i.e., they target each part of the text by construction. Thus, the answer distribution is less skewed. However, following this procedure, the annotators also tend to introduce more lexical overlap bias when writing questions after reading the source of the answers.

Additionally, Table 2 shows that the combination of the two features, word-overlap (WO), and reciprocal rank (RR), gives a strong baseline for all the datasets in consideration. This simple rule-based model ranks candidates according to the lexical overlap between question and candidates, and, in the case when two sentences have the same amount of overlapping words, it uses the reciprocal rank (RR) as a discriminator.

3 Related Work

Despite the importance of global structure, most state-of-the-art models (Severyn and Moschitti, 2016; He et al., 2015; Madabushi et al., 2018; Tay et al., 2018b; Garg et al., 2019) do not take the global structure of candidates into account. They use a point-wise approach to maximize the score of positive candidates, i.e., those containing the answer, and minimize the score of those not containing the answer. Most models treat ranking as a binary classification problem. Nevertheless, other methods have been studied, e.g., contrastive pair-wise and, more recently, list-wise approaches.

In the case of contrastive pair-wise training (Rao et al., 2016), given a question q, the loss maximizes the score of the model for a positive question candidate pair $(q, \hat{c}+)$ in contrast to the score of the negative pair $(q, \hat{c}-)$. This approach intrinsically balances the distribution of positive and negative examples in training and can improve the overall results. In particular, when paired with a hard negative sampling strategy. However, the comparison between the answer candidates is just performed at the score level; therefore, the model is still point-wise in terms of internal representation. The list-wise approach was proposed in a recent paper by Bian et al. (2017). The approach predicts the score for each question candidate pair individually, but it applies a softmax function on top of the set of scores, $s_1, s_2, ..., s_n = softmax(\phi(q, c_1), \phi(q, c_2), ..., \phi(q, c_n))$. This helps in providing stability in the training process. However, the model is agnostic of the global structure of the rank.

Most AS2 models have a way to identify the lexical overlap and, in particular, the semantic word-overlap between question and answer. The first neural models developed to solve the task (Yu et al., 2014; Severyn and Moschitti, 2015) directly added the lexical overlap feature in the model by concatenating it to the question-candidate representation or as a feature concatenated to the word embeddings of Convolutional Neural Network (CNN). Subsequent approaches, e.g., (Wang and Jiang, 2016; Bian et al., 2017), use a word-level attention mechanism to identify the semantic overlap between each word in the question and each word in the answer candidate. Despite obtaining better results than previous approaches, the computational cost of performing word-level attention and the aggregation steps to leverage the information extracted by the attention mechanism increases the computational cost of previous methods. More recent models, e.g., (Lai et al., 2019; Garg et al., 2019; Yoon et al., 2018), leverage contextualized word representation, e.g., pre-trained using BERT, ELMo, RoBERTa, etc. These approaches achieve state-of-the-art results for AS2, but they require significant computational power for both pre-training, fine-tuning, and testing on the final task.

4 Efficient Model for AS2

To build an efficient yet accurate model for AS2, it is crucial to leverage all the strong signals present in the dataset without increasing the complexity of the model itself. For this reason, we design a model as follows: (i) We build an efficient encoder to capture the question-candidate pair's lexical-overlap, i.e., our *Cosinet*. (ii) We add a recursive neural network on top of the question-candidate pairs to capture the original rank's global structure. (iii) We apply a global, list-wise, optimization approach to rank all the candidate pairs jointly.

4.1 Cosinet

The Cosinet has three building blocks: (i) a word-relatedness encoder that performs the cosine similarity between the word embeddings in the question and the answer (generating word relatedness features); (ii) similarly to (Severyn and Moschitti, 2016), the relational features are concatenated to the word embeddings and given in input to one layer of CNN, to create a representation for the question and candidate pair; and (iii) similarly to (Chen et al., 2017b), the information of the question and the candidate is combined at classification stage, by concatenating the vectors. That is, we use the component-wise multiplication and difference between question and answer vectors.

4.1.1 Word-Relatedness encoder

To encode the word-relatedness information, we first map the words in the question and the answer to their respective word embeddings. We then perform comparisons between all the embeddings in the question w_i^q and all the embedding of the answer w_j^c using the cosine similarity, $r_{i,j} = (w_i^q \cdot w_j^c)/(\|w_i^q\| \cdot \left\|w_j^c\right\|)$.

Instead of using the weighted sum of the embeddings as in standard attention, for each word in the question, we take the maximum relatedness score between its embedding and each word embedding of the candidate, i.e., $\mathbf{r}_i = max_j(r_{i,j})$. The same process is performed for each word in the answer. This value represents *how much a word is similar to the most similar word in the other text*. The value is concatenated to the word embedding of the question, i.e., $\hat{w}_i^q = [w_i^q; \mathbf{r}_i]$. A similar procedure is applied to the candidate to find $\mathbf{r}_j = max_i(r_{i,j})$, which is used to obtain the vector $\hat{w}_j^c = [w_j^c; \mathbf{r}_j]$. The two vectors, $\mathbf{r}_i$ and $\mathbf{r}_j$ are passed to the question-candidate encoder to create a pair representation. It is important to note that we keep the word embedding static during training, thus this operation does not cause much overhead during training as we do not need to back-propagate through it.

For this model, we use the Numberbatch (Speer and Lowry-Duda, 2017) embeddings since they are more accurate than unsupervised word embeddings, such as Glove (Pennington et al., 2014), which may introduce noisy or non-common-sense relations. In particular, Faruqui et al. (2016) showed that unsupervised word embeddings tend to cluster according to the frequency of words of the dataset used for training them. For this reason, Speer and Lowry-Duda (2017) adopted retrofitting. This technique aims at reducing the distance between word embeddings of entities that are related in a knowledge base, i.e., ConcepNet.

4.1.2 Question-Candidate encoder

Similarly to Severyn and Moschitti (2016), we encode the question and candidate independently using two single layers of CNN with a kernel size of 5 and global max pooling, producing two embeddings for the question q_e and the candidate c_e. These are then combined using their point-wise multiplication and their difference, i.e., $qc_e = [q_e \odot c_e; q_e - c_e]$.

4.2 Global optimization

Standard approaches take the pair representation qc_e and apply a feed-forward network that outputs the score s_i for the pair (q, c_i). However, this simple model cannot capture the inter-dependencies between the candidates for the question q, i.e., it does not capture the global structure of the rank.

In contrast, we use a Recurrent Neural Network applied on top of the qc_e representations for each c_i of a given question q, to leverage the global structure of the rank. The resulting contextual representations $\hat{qc}_e$ are passed to the feed-forward network (in our experiments, we use a single layer to produce the final

score). Finally, similarly to Bian et al. (2017), we apply a softmax function to the scores $s^1, .., s^n$ of all the n candidate answers of a given question. Then, we minimize the KL-divergence of the predicted probabilities and the normalized gold labels. The latter are normalized to generate a valid probability distribution over the candidates since there may be more than one answer sentence for each question. Bian et al. (2017) showed that this approach could improve the convergence speed of the model with respect to point-wise approaches.

5 Experiments

In these experiments, we first report the results of simple baselines and the state of the art for efficient and expensive models. Then, we present the results of our models carrying our comparison with the previous results.

5.1 Datasets

We used one dataset specific to AS2 and two datasets we adapted from MRC to AS2:

WikiQA Questions are randomly sampled from the Bing search engine logs. The candidate answers are the sentences that constitute the first paragraph of the related English Wikipedia article. Additionally, answers are concentrated in the first part of the paragraph.

SQuAD-sent For each question, the SQuAD dataset provides a paragraph and annotations for the exact answer span. To adapt SQuAD to the AS2 task, we split the paragraph into sentences using the SpaCy sentence tokenizer. We infer the sentence labels from the answer span labels, i.e., if a sentence contains the answer span, we labeled it as a positive example. Otherwise, we labeled it as a negative example. Since the SQuAD test set is not publicly available, we use the original validation set for testing and 10% of the questions of the training set for validation.

It should be noted that QNLI (the GLUE adaptation of SQuAD (Wang et al., 2018)) provides an even amount of positive and negative question/answer pairs, sampled from the SQuAD dataset, therefore removing many sentences. This creates a dataset of decontextualized sentences, which prevents us from exploiting the sequential structure. In contrast, we propose a dataset that maintains the original document structure. Finally, for all datasets, we remove the questions without answers and, we lowercase and tokenize[1] questions and passages to align with previous work.

We evaluate our models using two metrics: Precision at 1 (P@1) and Mean Reciprocal Rank (MRR). For SQuAD-sent, MAP is redundant since each paragraph contains, at most, one answer sentence. Additionally, SQuAD-sent exhibits a much more substantial lexical overlap between question and answer passages. This effect can be noted in Tab. 2: the simplest baseline that selects sentences in terms of word-overlap counts, i.e., counting the unique words that appear in both question and passage, achieves a P@1 of 65.48.

NQLA The Natural Question dataset uses question sampled from the Google search engine logs. The questions are given to the annotators together with the retrieved English Wikipedia page. The annotator is asked to select (i) a long answer, which is the smallest HTML bounding box containing all the information needed to answer the question, and (ii) a short answer (if available), that is, the actual answer to the question. In this work, for consistency with the other two datasets, we consider only paragraphs as long answers, removing tables and lists. As the latter requires a different semantic approach than the one typically used for free text. A paragraph is defined by the HTML bounding box <p>and <\p>.

The dataset has a similar answer distribution of the others, i.e., P@1 46.06% and MAP 57.30, even if the candidates are much longer (paragraphs). These results are impressive, considering that a Wikipedia page contains an average of 30.95 paragraphs (of 98.76 words). We note that most pages give essential information about an entity in the first paragraph, i.e., in the summary paragraph.

Similarly to SQuAD, the annotations for the test set of Natural Questions are not publicly available. Therefore, we used the official development set as our test set and a portion of the training set for validation.

[1]The tokenization is performed using spaCy v.2.1 `https://spacy.io/`

Model	MAP	MRR
Baselines		
RR	64.21	64.26
WO	51.02	51.24
WO+RR	68.25	69.43
Related Work w/o pre training		
(Tay et al., 2018a)	71.20	72.70
W&J 2016	74.33	75.45
W&J 2016†	72.38 ± 1.4	73.44 ± 1.5
(Sha et al., 2018)	74.62	75.76
(Bian et al., 2017)	**75.40**	**76.40**
Related Word with pre-training		
(Yoon et al., 2018)	83.40	84.80
(Lai et al., 2019)	85.70	87.20
(Garg et al., 2019)	92.00	93.30

Table 3: Related Work on WikiQA test-set. †We run the official implementation with different random seeds.

5.2 Models and parameters

In our experiments, we used two different encoder architectures: the newly proposed Cosinet and our re-implementation of the Compare-Aggregate (CA) architecture. The former uses static Numberbatch embeddings[2] of size 300; a convolution hidden layer of size 300, and a kernel of size 5. For the CA architecture, we use the standard parameters of the original paper, but in contrast with it, we keep the embedding static as we empirically found that it leads to similar results while having a lower number of trainable parameters. For the RNN and the LSTM, we used the same hidden size as the input, i.e., the double of the size of the convolutional operation hidden layer. For the Bidirectional variations, i.e., BiRNN and BiLSTM, we set the hidden size as half of the input size in each direction, resulting in a comparable number of parameters.

All the models were trained for three epochs using slanted triangular learning rate scheduling (Howard and Ruder, 2018) without early stopping. In the case of the point-wise models, we used Adam optimizer with a maximum learning rate set at 2e-3, whereas for the list-wise approaches, we used a learning rate of 2e-4. All the experiments are performed on an Nvidia GTX 1080 ti GPU and an Intel Core I9-7900X processor.

5.3 State-of-the-art Results

Table 3 shows the state-of-the-art results with respect to the WikiQA dataset. The first block reports the performance of the baselines; these models are computed using the simple features described in Section 2.1, i.e., the reciprocal rank (RR), the lexical overlap (WO), and their combination WO+RR. The latter achieves results comparable with baseline CNN architectures. The second block of results shows the performance of models from previous work that do not use pre-trained language models. The third block presents the results of models that use both pre-trained language models and transfer learning. In particular, Yoon et al. (2018) use ELMo and transfer learning on the QNLI dataset; Lai et al. (2019) use BERT and perform transfer learning on the QNLI dataset, and Garg et al. (2019) use RoBERTa large and perform transfer learning from the Natural Question dataset. We note that the MAP of efficient models ranges between 71% to 75%, while the MAP of expensive models ranges between 83% to 92%.

5.4 Our models

The state of the art has shown that pre-trained Transformer models can achieve up to 17 absolute points more than efficient approaches. Thus, the reader may wonder why we should even attempt to partially fill the gap with them. The answer is straightforward: in most cases, the cost of such models is prohibitive for industrial scenarios. Most importantly, we can still obtain the efficiency we require and keeping the cost low by applying sequential re-ranking. For example, we can use a fast classifier to select a subset

[2]https://github.com/commonsense/conceptnet-numberbatch

Model	RNN	MAP	MRR	params	train-time
Baselines					
RR	-	64.21	64.26	-	-
WO	-	51.02	51.24	-	-
WO + RR	-	68.25	69.43	-	-
Our Models					
Cosinet	-	70.95 ± 0.6	72.86 ± 0.7	904k	6 sec
Cosinet_{list}	-	71.22 ± 0.2	73.07 ± 0.3	904k	5.5 sec
Cosinet_{list}	RNN	74.78 ± 0.6	76.35 ± 0.6	1.17M	7.5 sec
Cosinet_{list}	BiRNN	$\mathbf{75.62} \pm 0.8$	$\mathbf{77.13} \pm 0.9$	1.12M	8.9 sec
Cosinet_{list}	LSTM	74.31 ± 0.8	75.78 ± 0.9	1.99M	7 sec
Cosinet_{list}	BiLSTM	75.32 ± 0.6	76.85 ± 0.5	1.81M	9.5 sec
CA	-	72.03 ± 1.6	73.39 ± 1.7	2.89M	19 sec
CA_{list}	-	71.43 ± 1.0	73.55 ± 1.0	2.89M	18 sec
CA_{list}	RNN	74.73 ± 1.0	76.35 ± 1.2	5.05M	20 sec
CA_{list}	BiRNN	74.97 ± 1.2	76.44 ± 1.2	4.87M	21 sec
CA_{list}	LSTM	74.82 ± 1.1	76.42 ± 1.2	11.53M	25 sec
CA_{list}	BiLSTM	74.27 ± 1.0	75.74 ± 1.1	10.81M	25 sec
BERT_{base}	-	81.32	82.50	110.00M	17 min 50 sec

Table 4: Model comparison on the WikiQA test-set.

of candidates, which will then be given in input to the expensive approach. The latter selects the final answer, just running on a small number of the candidate. Matsubara et al. (2020) shows that this approach can reduce the number of candidates by one order of magnitude without losing an accuracy point. In this perspective, the more accurate the fast reranker is, the higher the final accuracy will be.

Thus, our aim is to improve efficient models. Our approaches are shown in Table 4. Cosinet and CA are the standard point-wise approach, trained on all the data using a fixed batch size and binary cross-entropy (BCE). $Cosinet_{list}$ and CA_{list} are the same base architecture but trained with a listwise approach, with KL-Divergence loss on all the question-answer candidate pairs for the same question.

We note that: (i) The Cosinet architecture achieves comparable results with respect to the more complex CA while having much lower trainable parameters. (ii) BiRNN seems the best at (a) exploiting the rank structure, as it gets the higher results on $Cosinet_{list}$, and (b) outperforming the other models consistently. (iii) In contrast, for both Cosinet and CA, we found that there is no much statistical difference between RNN and LSTM.

Model	RNN	P@1	MRR	params	train-time
Baselines					
RR	-	30.55	53.53	-	-
WO	-	65.48	77.90	-	-
WO + RR	-	73.12	83.60	-	-
Our Models					
Cosinet		86.18 ± 0.2	91.81 ± 0.1	904k	1 min 47 sec
Cosinet_{list}		85.12 ± 0.1	91.16 ± 0.1	904k	8 min 10 sec
Cosinet_{list}	BiRNN	86.18 ± 0.2	91.97 ± 0.1	1.12M	12 min 30 sec
CA		85.71 ± 0.2	91.49 ± 0.1	2.89M	6min 30 sec
CA_{list}		85.17 ± 0.6	90.69 ± 1.0	2.89M	24 min 11 sec
CA_{list}	BiRNN	$\mathbf{86.32} \pm 0.3$	$\mathbf{92.05} \pm 0.2$	4.87M	28 min 30 sec
BERT_{base}	-	92.44	95.62	110.00M	6 hr 50 min

Table 5: Model comparison on the SQuAD sent test-set.

Similar findings can be derived from the results of the SQuAD dataset in Table 5. Our base architecture achieves comparable results with the more complex CA model. Moreover, adding the BiRNN to the model further improves the results. It is important to note that simple semantic matching models can achieve very high results on the SQuAD task. This because the lexical overlap feature is more prominent than the global rank feature. Therefore the primary impact is given by the joint model using BiRNN.

Finally, Table 6 shows the results derived using the bigger Natural Question dataset. Despite the

Model	RNN	MAP	MRR	params	train-time
Baselines					
RR	-	57.30	60.41	-	-
WO	-	38.09	39.57	-	-
WO + RR	-	53.98	56.45	-	-
Our Models					
Cosinet		69.74	72.69	904k	1 hr 13 min
Cosinet$_{list}$		68.16	71.06	904k	17 min
Cosinet$_{list}$	BiRNN	73.28	76.05	1.12M	34 min
CA		69.88	72.77	2.89M	5h 39min
CA$_{list}$		69.82	72.78	2.89M	2h
CA$_{list}$	BiRNN	**74.21**	**76.88**	4.87M	2h 10 min

Table 6: Model comparison on the NQLA test-set.

different nature of the data, i.e., the candidates are paragraphs rather than sentences, our proposed model improves the baselines, especially, when combined with the BiRNN. Overall, we note an improvement over CA of 4.5 absolute points, e.g., from 69.88 to 74.21.

5.5 Efficiency analysis

From the experiments on all three datasets, it is clear that the proposed Cosinet architecture is more efficient than CA since it has much fewer parameters and more efficient attention computation. On the small WikiQA dataset, the model takes up to 9.5 seconds to train, achieving results comparable with the best models that do not make use of pre-trained language models — in contrast, training BERT-base on the same dataset requires 17 min and 50 sec. Additionally, our model has roughly 100x less trainable parameters than BERT-base, effectively reducing the memory requirement of the model. The difference between the two models is evident when comparing the training time on the SQuAD dataset. The BiRNN-Cosinet trains in about 12 minutes, which is much lower than the 6 hours and 50 minutes needed to train BERT-base on the same task, and around half the time required by the CA architecture.

6 Conclusions

In this paper, we argue that by exploiting the intrinsic structure of the original rank and an effective word-relatedness encoder, we can achieve competitive results in comparison with state of the art, while retaining high efficiency. We first analyzed the structure of standard datasets, highlighting the importance of the original rank's global structure. Capitalizing on this, we propose a model that both exploits (i) the rank structure using a simple RNN and (ii) the standard word-relatedness features while preserving high efficiency. The model uses around 1M parameters depending on the configuration and achieves better results than the previous work that does not make use of computationally expensive pre-trained language models. Our model takes 9.5 seconds to train on the WikiQA dataset, i.e., very fast compared to the ~ 18 minutes required by a standard BERT-base fine-tuning. On SQuAD, this difference is even higher, i.e., minutes vs. hours required by Transformer-based models.

It should be stressed the fact that our very fast and accurate models are essential for at least two real-world application scenarios: (i) when the efficiency and computation cost does not allow for using pre-trained Transformer models; and (ii) in case of sequential re-ranking, our model can be positioned earlier in the re-ranking pipeline.

Recent work has also been devoted to distill cheaper versions of Transformer models (Sanh et al., 2019; Aguilar et al., 2020). These inevitably lose accuracy with respect to the original larger models. Our joint models can be potentially applied to them to limit their accuracy drop. Additionally, our Cosinet is still competitive in terms of speed with any distilled models, as none of them can ever have such a low number of parameters. This means that Cosinet can always be valuable in the first position of a sequential reranker, possibly followed by distilled versions of Transformer models.

References

Gustavo Aguilar, Yuan Ling, Yu Zhang, Benjamin Yao, Xing Fan, and Chenlei Guo. 2020. Knowledge distillation from internal representations. In *AAAI*, pages 7350–7357.

Payal Bajaj, Daniel Campos, Nick Craswell, Li Deng, Jianfeng Gao, Xiaodong Liu, Rangan Majumder, Andrew McNamara, Bhaskar Mitra, Tri Nguyen, et al. 2016. Ms marco: A human generated machine reading comprehension dataset. *arXiv preprint arXiv:1611.09268*.

Weijie Bian, Si Li, Zhao Yang, Guang Chen, and Zhiqing Lin. 2017. A compare-aggregate model with dynamic-clip attention for answer selection. In *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management*, pages 1987–1990. ACM.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017a. Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1870–1879.

Qian Chen, Xiaodan Zhu, Zhen-Hua Ling, Si Wei, Hui Jiang, and Diana Inkpen. 2017b. Enhanced lstm for natural language inference. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1657–1668.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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)*, pages 4171–4186.

Manaal Faruqui, Yulia Tsvetkov, Pushpendre Rastogi, and Chris Dyer. 2016. Problems with evaluation of word embeddings using word similarity tasks. *arXiv preprint arXiv:1605.02276*.

Siddhant Garg, Thuy Vu, and Alessandro Moschitti. 2019. TANDA: transfer and adapt pre-trained transformer models for answer sentence selection. *CoRR*, abs/1911.04118.

Hua He, Kevin Gimpel, and Jimmy Lin. 2015. Multi-perspective sentence similarity modeling with convolutional neural networks. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 1576–1586.

Jeremy Howard and Sebastian Ruder. 2018. Universal language model fine-tuning for text classification. *arXiv preprint arXiv:1801.06146*.

Bernhard Kratzwald and Stefan Feuerriegel. 2018. Adaptive document retrieval for deep question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 576–581.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Matthew Kelcey, Jacob Devlin, Kenton Lee, Kristina N. Toutanova, Llion Jones, Ming-Wei Chang, Andrew Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association of Computational Linguistics*.

Tuan Lai, Quan Hung Tran, Trung Bui, and Daisuke Kihara. 2019. A gated self-attention memory network for answer selection. 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 5955–5961.

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. *arXiv preprint arXiv:1907.11692*.

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.

Yoshitomo Matsubara, Thuy Vu, and Alessandro Moschitti. 2020. Reranking for efficient transformer-based answer selection. In *Proceedings of the 42th international ACM SIGIR conference on research and development in information retrieval*.

Rodrigo Nogueira and Kyunghyun Cho. 2019. Passage re-ranking with bert. *arXiv preprint arXiv:1901.04085*.

Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. Glove: Global vectors for word representation. In *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pages 1532–1543.

Matthew E. Peters, Mark Neumann, Mohit Iyyer, Matt Gardner, Christopher Clark, Kenton Lee, and Luke Zettlemoyer. 2018. Deep contextualized word representations. In *Proc. of NAACL*.

Yifan Qiao, Chenyan Xiong, Zhenghao Liu, and Zhiyuan Liu. 2019. Understanding the behaviors of bert in ranking. *arXiv preprint arXiv:1904.07531*.

Alec Radford, Jeffrey Wu, Rewon Child, David Luan, Dario Amodei, and Ilya Sutskever. Language models are unsupervised multitask learners.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. *arXiv preprint arXiv:1606.05250*.

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. ACM.

Matthew Richardson, Christopher JC Burges, and Erin Renshaw. 2013. Mctest: A challenge dataset for the open-domain machine comprehension of text. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 193–203.

Victor Sanh, Lysandre Debut, Julien Chaumond, and Thomas Wolf. 2019. Distilbert, a distilled version of bert: smaller, faster, cheaper and lighter. *arXiv preprint arXiv:1910.01108*.

Aliaksei Severyn and Alessandro Moschitti. 2015. Learning to rank short text pairs with convolutional deep neural networks. In *Proceedings of the 38th international ACM SIGIR conference on research and development in information retrieval*, pages 373–382. ACM.

Aliaksei Severyn and Alessandro Moschitti. 2016. Modeling relational information in question-answer pairs with convolutional neural networks. *arXiv preprint arXiv:1604.01178*.

Lei Sha, Xiaodong Zhang, Feng Qian, Baobao Chang, and Zhifang Sui. 2018. A multi-view fusion neural network for answer selection. In *AAAI*.

Luca Soldaini and Alessandro Moschitti. 2020. The cascade transformer: an application for efficient answer sentence selection. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 5697–5708, Online, July. Association for Computational Linguistics.

Robert Speer and Joanna Lowry-Duda. 2017. Conceptnet at semeval-2017 task 2: Extending word embeddings with multilingual relational knowledge. In *Proceedings of the 11th International Workshop on Semantic Evaluation (SemEval-2017)*, pages 85–89.

Emma Strubell, Ananya Ganesh, and Andrew McCallum. 2019. Energy and policy considerations for deep learning in nlp. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, pages 3645–3650.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018a. Hyperbolic representation learning for fast and efficient neural question answering. In *Proceedings of the Eleventh ACM International Conference on Web Search and Data Mining*, pages 583–591. ACM.

Yi Tay, Luu Anh Tuan, and Siu Cheung Hui. 2018b. Multi-cast attention networks for retrieval-based question answering and response prediction. *arXiv preprint arXiv:1806.00778*.

Shuohang Wang and Jing Jiang. 2016. A compare-aggregate model for matching text sequences. *arXiv preprint arXiv:1611.01747*.

Mengqiu Wang, Noah A Smith, and Teruko Mitamura. 2007. What is the jeopardy model? a quasi-synchronous grammar for qa. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*.

Alex Wang, Amanpreet Singh, Julian Michael, Felix Hill, Omer Levy, and Samuel Bowman. 2018. Glue: A multi-task benchmark and analysis platform for natural language understanding. In *Proceedings of the 2018 EMNLP Workshop BlackboxNLP: Analyzing and Interpreting Neural Networks for NLP*, pages 353–355.

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.

Seunghyun Yoon, Joongbo Shin, and Kyomin Jung. 2018. Learning to rank question-answer pairs using hierarchical recurrent encoder with latent topic clustering. In *Proceedings of NAACL-HLT*, pages 1575–1584.

Lei Yu, Karl Moritz Hermann, Phil Blunsom, and Stephen Pulman. 2014. Deep learning for answer sentence selection. *arXiv preprint arXiv:1412.1632*.