DyReF: Extractive Question Answering with Dynamic Query Representation for Free

Urchade Zaratiana[1,2*], **Niama El Khbir**[2], **Pierre Holat**[1,2],
Nadi Tomeh[2], **Thierry Charnois**[2]
[1]FI Group, [2]LIPN, Université Sorbonne Paris Nord - CNRS UMR 7030

Abstract

Extractive QA is an important NLP task with numerous real-world applications. The most common method for extractive QA is to encode the input sequence with a pretrained Transformer such as BERT, and then compute the probability of the start and end positions of span answers using two leaned query vectors. This method has been shown to be effective and hard to outperform. However, the query vectors are static, meaning they are the same regardless of the input, which can be a challenging issue in improving the model's performance. To address this problem, we propose DyReF (Dynamic Representation for Free), a model that dynamically learns query vectors for free, i.e. without adding any parameters, by concatenating the query vectors with the embeddings of the input tokens of the Transformer layers. In this way, the query vectors can aggregate information from the source sentence and adapt to the question, while the representations of the input tokens are also dependent on the queries, allowing for better task specialization. We demonstrate empirically that our simple approach outperforms strong baseline in a variety of extractive question answering benchmark datasets. The code is publicly available at https://github.com/urchade/DyReF.

1 Introduction

Extractive QA [Rajpurkar et al., 2016] is a challenging NLP task with the goal of predicting the exact position of the answer spans given a document and a question as input. Early deep learning based ExQA models were very specialized and heavily engineered. These models are mostly based on LSTMs and attention mechanisms [Bahdanau et al., 2015]. The arrival of BERT [Devlin et al., 2019] and pre-trained language models transformed the domain by introducing a simple yet effective approach. It consists of concatenating the input question and text passage, then computing the start and end of the answer spans using learned query parameters [Devlin et al., 2019]. This approach has benefited from the Transformer-based language model pretraining [Devlin et al., 2019, Liu et al., 2019, Joshi et al., 2020], which has enabled little architecture engineering to obtain state-of-the-art results. Since its introduction, this objective has been widely adopted and is still the dominant approach for extractive QA.

However, despite their high performance, we argue that such methods remain suboptimal since the query vectors used to compute the start and end distributions are static, i.e., they are independent of the input sequence. We propose DyReF, a novel method to tackle this problem, which extends these models by allowing the queries to dynamically aggregate information from the input sequence to better answer the question. To do so, we feed the initial query representations in the Transformer model's input along with the input sequences. This approach allows for (1) interaction between the queries and the input sequence, (2) modeling the interdependence between the queries since the

*Correspondence to: zaratiana@lipn.fr

Workshop on Transfer Learning for Natural Language Processing, 2022, New Orleans.

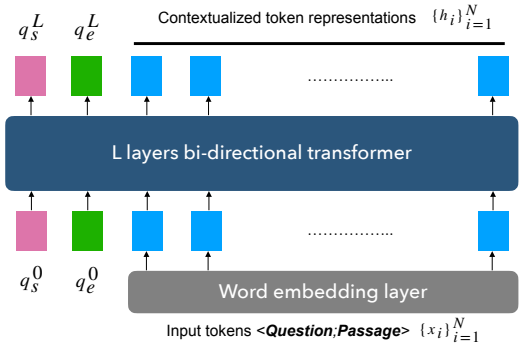

Figure 1: **Illustration of our framework:** DyReF. Start and end query vectors ($\mathbf{q}_s$ and $\mathbf{q}_e$) are appended to the input sequence and are then passed to the L-layers pre-trained transformer model to obtain dynamic query vectors and specialized (or "query-aware") token embeddings. For the *vanilla*/standard ExQA model, $\mathbf{q}_s$ and $\mathbf{q}_e$ are statics; they remain the same for different inputs.

positions of the start and end of the answer are mutually dependent, and (3) better specialization of the token representation by using information from the queries. We draw all these advantages from the Transformer attention mechanism. Another main advantage of our approach is maintaining the same number of parameters as the model with static query representation while obtaining better performance.

We conduct extensive experiments on several extractive Question Answering benchmarks, including SQuAD [Rajpurkar et al., 2016] and MRQA datasets [Fisch et al., 2019]. Experimental results demonstrate that our approach, DyReF, consistently improves the performance over the widely used baseline on large and small datasets. In particular, we show that our approach outperforms the baseline model by a large margin in few-shot scenarios.

The rest of this paper is organized as follows. In the background section we provide some context for understanding the baseline model and the primary motivation behind our approach. Then we present our proposed model in detail, followed by our experimental setup, results, and further experimental analysis, as well as an overview of related work. The last section concludes this paper.

2 Background

2.1 *Vanilla* ExQA

Pre-trained Transformer encoders have been extremely useful in recent NLP research, enabling significant improvements across a wide range of tasks thanks to self-supervised pre-trainings on large corpora. For example, at the time of its publication, BERT outperformed all extractive question answering benchmarks by a wide margin compared to previous heavily engineered LSTMs and attention-based architectures such as BidAF [Seo et al., 2016].

This section outlines the mainstream approach to Extractive Question Answering tasks, which we refer to as *Vanilla* ExQA. It is typically performed by feeding a pre-trained Transformer language model such as BERT the input sequence $\{x_i\}_{i=1}^N$ which is the concatenation of the tokenized question Q and the passage P containing the answer. The Transformer produces a set of contextualized token representations $\{\mathbf{h}_i\}_{i=1}^N \in \mathbb{R}^d$, d being the embedding dimension of the model. Then, to compute the probability of the *start* and *end* position of the answer span, the following estimators are used:

$$p(start = i|Q, D) = \frac{\exp(\mathbf{q}_s^T \mathbf{h}_i)}{\sum_{i'=1}^N \exp(\mathbf{q}_s^T \mathbf{h}_{i'})} \qquad p(end = j|Q, D) = \frac{\exp(\mathbf{q}_e^T \mathbf{h}_j)}{\sum_{j'=1}^N \exp(\mathbf{q}_e^T \mathbf{h}_{j'})} \quad (1)$$

Where $\mathbf{q}_s$ and $\mathbf{q}_e \in \mathbb{R}^d$ are the *start* and *end* queries respectively. These queries are initialized randomly and learned during the training of the model.

The training objective is to minimize the sum of the negative log-likelihood of the correct start and end positions $(\hat{i}, \hat{j})$:

$$\mathcal{L} = -\log p(\text{start} = \hat{i}|Q, P) - \log p(\text{end} = \hat{j}|Q, P) \tag{2}$$

This training objective assumes independence between the start and end of the answer spans, but in practice, it achieves more stable and more reliable results than joint objective approaches [Fajcik et al., 2021]. This approach was first proposed by Devlin et al. [2019] and is now adopted by most Transformer-based ExQA models [Liu et al., 2019, Joshi et al., 2020].

3 Our approach: DyReF

3.1 Main idea

The learned query vectors $\mathbf{q}_s$ and $\mathbf{q}_e$ of the *vanilla* approach are static, i.e., they are shared by all input sentences, and as a result, are context-insensitive. We suppose that using these static queries is a bottleneck for performance improvement, so we propose to extend this strategy by allowing the queries to dynamically aggregate information from the input sequence, allowing the model to efficiently adapt to the context.

DyReF, as illustrated in Figure 1, first computes word embeddings $\{\mathbf{h}_i^0\}_{i=1}^N$ of the input sequence $\{x_i\}_{i=1}^N$, which corresponds to the concatenation of the tokenized question and the passage containing the span answer. Then, it appends the initial start query $\mathbf{q}_s^0$ and end query $\mathbf{q}_e^0$ to the input sequence embeddings producing $\{\mathbf{q}_s^0, \mathbf{q}_e^0, \mathbf{h}_1^0, \ldots, \mathbf{h}_N^0\}$ which is feed to a series of L Transformer layers, $\texttt{Trans}_L$, to obtain the representations of the queries and the input tokens:

$$[\mathbf{q}_s^L, \mathbf{q}_e^L, \mathbf{h}_1, \ldots, \mathbf{h}_N] = \texttt{Trans}_L([\mathbf{q}_s^0, \mathbf{q}_e^0, \mathbf{h}_1^0, \ldots, \mathbf{h}_N^0]) \tag{3}$$

Before training the model, we randomly initialize the first queries $\mathbf{q}_s^0$ and $\mathbf{q}_e^0$ using the normal distribution, with a mean of 0 and a standard deviation of 0.02. We also initialize both the word embeddings and the Transformer layers using a pre-trained Transformer-based language model.

Furthermore, to compute the probability of the start and the end answer spans, we use the estimators of equation **??**, replacing $\mathbf{q}_s$ and $\mathbf{q}_e$ by the final queries $\mathbf{q}_s^f := \mathbf{q}_s^L$, and $\mathbf{q}_s^f := \mathbf{q}_e^L$, the output queries of the L-th Transformer layer. In the next subsection, we propose more robust alternatives for computing $\mathbf{q}_s^f$ and $\mathbf{q}_e^f$.

Benefits The main advantage of our approach is that it allows interaction between queries and input tokens through the attention mechanism, unlike the *vanilla* model that uses static queries. More precisely, query representations become dependent on token representations, allowing them to adapt to each specific input to better answer a question. Furthermore, the *start* and the *end* queries become dependent on each other, which is convenient since the position of the *start* and *end* of answer span are indeed dependent. Our approach makes the task of the model easier since the span answer can be located at multiple places in the input passage, and so collaboration between the start and end queries is crucial. In addition, the input representations also receive information from the queries which can be beneficial to the model.

3.2 Query representation strategies

In our work, we try several alternatives to compute the final queries, $\mathbf{q}_t^f$ where $t \in \{s, e\}$.

Last layer The most straightforward approach is to take the last layer's query representations as the final representations:

$$\mathbf{q}_t^f := \mathbf{q}_t^L \tag{4}$$

This setup supposes that the queries from the last layers contain sufficient information for accurately predicting the span answer.

***Mean* pooling** The assumption here is that all intermediate queries may contain useful information. Combining these queries to compute the final ones could be more robust than only using the last ones. We propose to present the final query as the average of all queries from the L-layers:

$$\mathbf{q}_t^f := \frac{1}{L}\sum_{l=1}^{L}\mathbf{q}_t^l \tag{5}$$

***Max* pooling** The final queries are calculated here by taking the maximum value for each representation dimension across all layers:

$$\mathbf{q}_t^f := \texttt{MAX}([\mathbf{q}_t^1, \ldots, \mathbf{q}_t^L]) \tag{6}$$

In contrast to *Mean* pooling, which assumes they all have the same importance, the max-pooling operation allows the model to dynamically select the most salient information from each layer.

4 Experimental Setup

4.1 Data

For our experiments, we employ widely used benchmark extractive question-answering datasets, which include SQuAD [Rajpurkar et al., 2016], HotpotQA [Yang et al., 2018], NewsQA [Trischler et al., 2016], TriviaQA [Joshi et al., 2017] and Natural Questions [Kwiatkowski et al., 2019]. These datasets come from various sources such as Wikipedia articles, news articles and Web snippets. In each of these datasets, a triple of "Question, Passage, Answer" is provided, and the task is to predict the precise position of the answer (start and end positions) given the question and the passage that contains the answer. The dataset statistics are detailed in Table 1 of the MRQA shared task paper [Fisch et al., 2019].

4.2 Evaluation metrics

F1 scores and Exact Matching (EM) scores are the basic metrics used to evaluate the extractive question answering task. The F1 measures the average overlap between the prediction and the ground truth answer by considering both the prediction and the ground truth as bags of tokens and computing their macro-averaged F1 score. The EM measures the percentage of predictions that matches *exactly* any one of the ground truth answers [Rajpurkar et al., 2016].

4.3 Hyperparameters

We did not perform a hyperparameter search in this work. Instead, we adopted standard configurations existing in the literature for better reproducibility. Specifically, we adapted the existing codebase from [Ram et al., 2021] and [Joshi et al., 2020] GitHub repositories, which themselves use the default HuggingFace Transformers library [Wolf et al., 2020] hyperparameters configuration. We employed Adam optimizer [Kingma and Ba, 2015] with a learning rate of 3.10^{-5}, where a warm-up stage is set for the first 10% of the steps, then decay the learning rate linearly for the rest of the training steps. We set the batch size to 12, and train for a maximum of 5 epochs for full-sized datasets. We train the models up to either 2500 steps or 10 epochs in few-shot settings. We borrowed the baseline *vanilla* QA implementation from [Ram et al., 2021] repository. We used PyTorch [Paszke et al., 2019] to implement the models and load the pre-trained models from the Transformers library [Wolf et al., 2020]. We trained all models in a server with V100 GPUs. In total, all the experiments described in this paper as well as some preliminary experiments required about 2500 GPU hours.

5 Results

In this section, we present our main experimental results in Table 1, comparing different variants of `DyReF` to the *vanilla* ExQA model. For the different approaches, we used either BERT or SpanBERT for token representation.

Enc	Models	SQuAD		HotpotQA		TriviaQA		NewsQA		NaturalQs		Avg.	
		EM	F1	EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
SpanBT	*Vanilla*	82.96	90.56	63.89	79.83	72.42	77.05	52.66	67.72	65.26	77.38	67.48	78.58
	Last	83.34	90.88	**64.75**	80.51	72.23	77.26	52.63	67.97	65.98	78.54	67.79	79.03
	Mean	83.6	**91.08**	64.50	80.28	71.98	76.95	**52.94**	**68.66**	**66.29**	**78.56**	**67.86**	79.11
	Max	83.03	**91.08**	64.56	**80.55**	**72.49**	**77.39**	52.4	68.27	65.96	78.44	67.69	**79.15**
BERT	*Vanilla*	**79.65**	87.42	60.41	76.63	64.53	69.85	48.52	63.25	62.39	74.76	63.10	74.38
	Last	79.27	**87.83**	**61.04**	**77.42**	64.32	69.18	47.72	63.84	62.91	**75.41**	63.05	74.74
	Mean	79.30	87.54	60.72	76.92	**65.36**	**70.63**	**49.76**	**64.91**	62.82	74.95	**63.59**	**74.99**
	Max	79.18	87.65	60.14	76.73	64.55	69.72	49.10	64.66	**62.96**	74.95	63.19	74.74

Table 1: **Main results**. We compare variants of DyReF (*Last* (Eq. 4), *Mean* (Eq. 5) and *Max*) to the *vanilla* ExQA approach (Eq. 1). For each approach (using BERT [Devlin et al., 2019] or SpanBERT [Joshi et al., 2020] as token encoder) and for each dataset, we report the F1 score and the EM score.

BERT We first analyze the obtained results when using BERT [Devlin et al., 2019] as a token encoder. We observe that DyReF with *Mean* pooling is the representation approach that achieves the best F1 performance on average. It obtains the highest F1 score on three datasets, namely SQuAD, TriviaQA, and NewQA datasets. Moreover, *Max* and *Last* have about the same performance still outperform the *vanilla* ExQA model in terms of F1 score and EM.

SpanBERT In terms of global performance, using SpanBERT [Joshi et al., 2020] always outperforms the BERT-based model. In fact, SpanBERT's pre-training is highly optimized for span-related tasks, such as coreference resolution and extractive question answering. This shows the importance of pre-training for the downstream tasks. Regarding the relative performance of the different approaches using SpanBERT, they all outperform the *vanilla* model in terms of average F1 and EM scores, except for a few experiments with TriviaQA and NewsQA where the *Mean* and the *Max* representation score lower than the *vanilla* ExQA. We observe that similarly to BERT, the best representation approach using SpanBERT is the *Mean*. It gets the highest averaged EM score and the second best-averaged F1. Moreover, It obtains the highest F1 score on SQuAD, NewsQA, and Natural Questions datasets.

Overall For both BERT and SpanBERT, we conclude that the *Mean* approach is the most robust, achieving the best average performance for the two token representation approaches. Finally, all our proposed variants outperform the *vanilla* ExQA model on average, whether with BERT or SpanBERT.

6 Few-shot performance

We conduct few-shot experiments and report the results in Table 2. In all these experiments, we used SpanBERT [Joshi et al., 2020] as the token encoder representation since we observed previously that it always outperforms the BERT Devlin et al. [2019] representation in full-sized experiments and by a large margin sometimes. We used dataset sizes ranging from 128 to 1024 data points, employing the same splits as Ram et al. [2021], available in their GitHub repository.

***Vanilla* vs.** DyReF In the few-shot scenarios, we observe that our models perform significantly better than the vanilla approach both in terms of F1 and EM scores. For instance, on the SQuAD dataset using 128 data points, the *Max* representation exceeds the *vanilla* model by 8.47 and 6.63 in terms of EM and F1 respectively. However, while the performance gap is large on small data regimes, it becomes smaller as the size of the training datasets increases. Overall, these results show that dynamic queries are beneficial when data is scarce, thus our models could be helpful for future works on domain-specific ExQa tasks where annotating data can be time-consuming, expensive, and requires expertise such as legal or biomedical domains.

Comparing variants We compare here our different variants of DyReF. We observe that all our different approaches perform similarly across the datasets, but we note that the *Max* approach performs best in many scenarios; in fact, it scores best on 3 of the 5 datasets when trained with 256 and 512 data points, and also obtains strong results on other dataset sizes.

Train size	Models	SQuAD		HotpotQA		TriviaQA		NewsQ		NaturalQs	
		EM	F1	EM	F1	EM	F1	EM	F1	EM	F1
128	*Vanilla*	43.51	54.83	27.64	41.16	15.43	20.23	18.02	29.89	20.79	29.21
	Last	49.97	59.48	28.65	40.63	18.14	23.64	**24.22**	**36.55**	**27.14**	**35.33**
	Mean	50.76	60.66	27.77	39.55	**24.38**	**30.60**	24.10	35.85	25.67	33.76
	Max	**51.98**	**61.46**	**33.81**	**45.73**	21.54	27.43	22.82	33.99	26.78	34.23
256	*Vanilla*	55.94	65.74	37.87	53.23	23.78	28.49	25.07	35.80	32.83	41.87
	Last	61.86	71.71	39.08	53.41	30.03	37.24	**32.10**	**45.64**	34.79	44.52
	Mean	**62.36**	**71.87**	39.48	53.22	32.95	38.64	31.70	45.61	36.12	45.43
	Max	61.14	71.08	**40.64**	**55.08**	**33.98**	**40.53**	30.39	44.56	**37.07**	**46.91**
512	*Vanilla*	60.69	69.72	43.58	58.70	40.03	45.39	30.31	43.24	38.73	48.36
	Last	66.91	76.71	46.25	61.17	45.23	52.62	35.04	49.94	42.50	53.28
	Mean	67.39	76.97	46.84	61.91	**48.32**	**54.58**	35.66	50.24	42.79	53.24
	Max	**67.55**	**77.24**	**46.94**	**62.27**	46.86	52.59	**36.85**	**52.26**	**43.75**	**54.51**
1024	*Vanilla*	64.72	74.01	46.87	62.54	46.28	51.87	37.20	50.61	43.72	53.42
	Last	69.39	78.72	49.89	65.50	50.66	56.94	40.91	55.88	**49.89**	**61.49**
	Mean	70.29	79.95	50.52	66.13	**51.65**	**57.80**	40.24	55.85	49.12	60.07
	Max	**70.72**	**80.35**	**50.97**	**66.40**	50.94	56.77	**40.98**	**56.71**	49.52	61.15

Table 2: **Few-shot performance**. This table reports our few-shot experiments. We evaluate all the datasets by training on different training sizes, ranging from 128 to 1024 data points. For all the experiments, we employ SpanBERT for token representation as it consistently outperforms BERT.

Summary Overall, we find that using dynamic query representations, i.e., `DyReF`, is crucial in a few datapoints scenarios as it provides good results over the widely used *vanilla* model. In fact, although the difference is not significant under full data settings, `DyReF` outperforms *vanilla* by a significant margin when using little training data. Furthermore, we remark that the *Max* variant provides the strongest and most consistent results across the datasets.

7 Attention analysis

The attention mechanism plays a crucial role in our work since it allows the queries to interact with each other and with the input tokens. In this section, we perform a detailed analysis of the contribution of the attention mechanism for a better understanding of the model. To do so, we carry out two studies: 1) in the first study, we compare different attention masks to identify the most relevant query-query interactions and token-query interactions; and 2) in the second study, we visualize the attention map to determine the area of the input that the queries pay the most attention to, which may help to explain and better comprehend the model.

7.1 Attention mask variants

Motivation In `DyReF`, the queries can interact among each other as well as with the input tokens thanks to the attention mechanism. Our approach can therefore model the dependency between queries and input sequence elements by learning to aggregate useful information through the Transformer layers to better answer a question. In this section, we propose to analyze some variants of `DyReF` by trying different attention masks for a better understanding of their importance for our model. The main idea is to mask some interactions in order to detect their influence on the model performance in terms of F1 score and EM metrics.

Masking strategies We propose to analyze three variants of our model, illustrated in Figure 2. All our studied variants use dynamic queries but employ different attention masks between the queries: the first variant, *Bidirectional*, allows full attention between queries; then *Causal* allows the end query to attend the start query but not vice versa, and finally, *Independent* completely mask attention between queries making them independent.

self-att	EM	F1
a) Static (*vanilla*)	83.08 ± 0.09	90.64 ± 0.10
Dynamic		
+ b) *Full*	$\mathbf{83.43 \pm 0.18}$	$\mathbf{91.04 \pm 0.04}$
+ c) *Bidirectional*	83.37 ± 0.26	$\mathbf{91.04 \pm 0.05}$
+ d) *Causal*	$\mathbf{83.43 \pm 0.07}$	90.99 ± 0.07
+ e) *Independent*	83.03 ± 0.08	90.88 ± 0.01

Table 3: **Different masking variants for the self-attention layers**. This study is performed on SQuAD dataset using *Mean* and SpanBERT for token representation. Results are averaged across three random seeds.

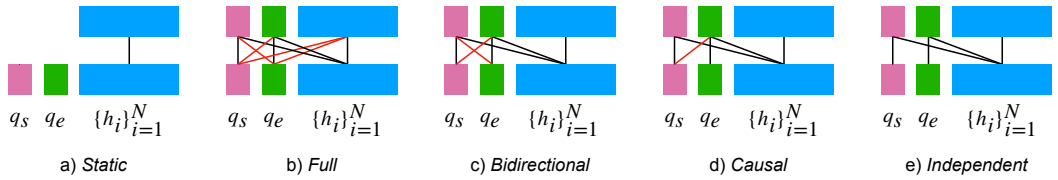

Figure 2: **Different attention masking for** DyReF. a) *Static* is the one employed by the *vanilla* ExQA, in which the queries do not depend on the input. b) *Full* is our main model where both all the queries and all the tokens attend to each other. c) *Bidirectional* allow full interaction between the queries and remains dynamic (depends on input tokens), however, the input tokens do not attend to the queries. d) *Causal* is the same as the bidirectional except that the start query does not attend to the end query. e) *Independent* also depends on the input sequence but queries are not attending to each other, they are completely independent.

Results The results, reported in Table 3, show that the best results both in terms of F1 and EM metrics are achieved by the *Full* attention model. However, both *Causal* and *bidirectional* obtain competitive results which demonstrate that interaction between the queries are important since the *Independent* approach has lowest score.

7.2 Attention visualization

In addition to the attention mask study, we visualize the attention score of the transformer layer of DyReF. The main motivation for this analysis is to see which part of the input sequence the query vectors interact with the most, which gives us more insight into understanding our model. The visualization is shown in Figure 3. For the visualizations, we employed a trained QA model on the SQuAD datasets using SpanBERT [Joshi et al., 2020] for token representations using *Last* for query representation. We observe a similar behavior when visualizing the attention map with *Mean* and *Max*.

Method Since SpanBERT [Joshi et al., 2020] has 12 layers and each layer contains 12 attention heads, this results in 12×12 attention matrices. To ease the interpretation, we decide to aggregate all the attention matrices into a single one: to do so, we simply employed a 2D max pooling over the head and layer dimension of the staked attentions matrices resulting in only one attention matrix of dimension $(2 + N) \times (2 + N)$ where N is the length of the input sequence and 2 the number of queries. Finally, we only visualize the query-query and query-token attention (i.e, we drop token-token interaction) resulting in a attention map of $2 \times (2 + N)$ in figure 3.

Analysis The resulting attention visualization is shown in Figure 3. We conduct the study using two different inputs to assess the consistency of our model: the two examples share the same passage but have different questions and answers. This visualization shows that the queries have already the knowledge of the answer span position even before making the final prediction, which may be the reason DyReF is more robust than the *vanilla* approach.

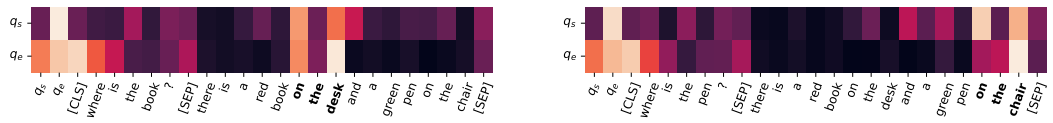

Figure 3: **Attention visualization**. This figure shows to which tokens the start and end queries ($\mathbf{q}_s$ and $\mathbf{q}_e$) are attending. For this study, we a trained QA model on the SQuAD dataset using SpanBERT as token representation and *Last* for query representations. To create the attention map, we elementwise-max the attention values over all the 12 attention heads and 12 layers of the models. For the two examples, we employ the sample passage but different questions to evaluate the consistency of the attention. We put on boldface the tokens of the answer spans, for the first example (left) the question is "Where is the book ?" and the answer is "on the desk". This question/passage pair example is taken from Cui et al. [2022].

8 Related Works

ExQA models Extractive question answering is an important NLP task with numerous real-world applications. The advance in this field has been mainly due to the availability of large-scale annotated datasets such as SQuAD [Rajpurkar et al., 2016]. Early deep learning based ExQA models, such as BidAF [Seo et al., 2016], Match-LSTM [Hu et al., 2017] and QaNet [Yu et al., 2018] among others [Shen et al., 2016, Wang et al., 2017, Cui et al., 2017, Dhingra et al., 2017, Clark and Gardner, 2018, Hu et al., 2018, Liu et al., 2018, Huang et al., 2018], were very specialized and heavily engineered. The introduction of BERT [Devlin et al., 2019] has dramatically transformed the ExQA paradigm by introducing a simple yet effective approach: concatenating the input question and text passage and computing the start and end span answer using learned query parameters [Devlin et al., 2019]. In this paper we call this model *vanilla* ExQA. Since its introduction, this objective is still the dominant approach for extractive QA [Liu et al., 2019, Yang et al., 2019, Joshi et al., 2020, Yamada et al., 2020, He et al., 2021, Yasunaga et al., 2022]. In this paper, we extend the *vanilla* ExQA by employing dynamic query representation while keeping the same parameters number as the *vanilla* ExQA and as well as a negligible computational cost. Furthermore, frameworks such as Splinter [Ram et al., 2021] and ReasonBERT [Deng et al., 2021] have recently been proposed and have produced state-of-the-art results on extractive question answering. However, the main contribution of these models was the introduction of effective pretext tasks for improving ExQA models and thus require additional self-supervised pre-training. Finally, our work is similar to the recently proposed *DyREx* [Zaratiana et al., 2022], which also uses dynamic queries, however, it requires the addition of extra parameters to the model whereas in our case the dynamic queries are obtained for free.

Relation to prefix tuning `DyReF` is inspired by *prefix tuning* [Li and Liang, 2021] which keeps the pre-trained models fixed and only updates some prefix token parameters appended to the input token embeddings before transformer layers. However, our approach differs on some points: (1) first, we use bidirectional encoders instead of left-to-right language models, and we tune the whole model parameters instead of only tuning the prefixes (i.e the queries); by only tuning the queries and freezing the model weights, our model suffers from under-fitting resulting in poor performance; (2) also, the main goal of *prefix tuning* is parameter efficient fine-tuning while the core motivation of our approach is to obtain dynamic representations of the queries for extractive question answering purpose.

9 Conclusion

In this paper, we presented `DyReF`, a new approach to ExQA using learnable dynamic queries. We show that our approach outperforms the *vanilla* model on several datasets without increasing the number of parameters, and maintaining approximately the same computational speed. For future works, it would be interesting to adapt our proposed architecture for tasks involving the extraction of multiple spans such as keyphrase extraction and multi-span question answering [Li et al., 2022].

Acknowledgments

This work is partially supported by a public grant overseen by the French National Research Agency (ANR) as part of the program Investissements d'Avenir (ANR-10-LABX-0083). This work was granted access to the HPC/AI resources of [CINES/IDRIS/TGCC] under the allocation 20XX-AD011013682 made by GENCI.

References

Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. Neural machine translation by jointly learning to align and translate. *CoRR*, abs/1409.0473, 2015.

Christopher Clark and Matt Gardner. Simple and effective multi-paragraph reading comprehension. *ArXiv*, abs/1710.10723, 2018.

Yiming Cui, Z. Chen, Si Wei, Shijin Wang, Ting Liu, and Guoping Hu. Attention-over-attention neural networks for reading comprehension. *ArXiv*, abs/1607.04423, 2017.

Yiming Cui, Wei-Nan Zhang, Wanxiang Che, Ting Liu, Zhigang Chen, and Shijin Wang. Multilingual multi-aspect explainability analyses on machine reading comprehension models. *iScience*, 25 (5):104176, 2022. ISSN 2589-0042. doi: https://doi.org/10.1016/j.isci.2022.104176. URL `https://www.sciencedirect.com/science/article/pii/S2589004222004461`.

Xiang Deng, Yu Su, Alyssa Lees, You Wu, Cong Yu, and Huan Sun. Reasonbert: Pre-trained to reason with distant supervision. In *EMNLP*, 2021.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. BERT: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, Minneapolis, Minnesota, June 2019. Association for Computational Linguistics. doi: 10.18653/v1/N19-1423. URL `https://aclanthology.org/N19-1423`.

Bhuwan Dhingra, Hanxiao Liu, Zhilin Yang, William W. Cohen, and Ruslan Salakhutdinov. Gated-attention readers for text comprehension. *ArXiv*, abs/1606.01549, 2017.

Martin Fajcik, Josef Jon, Santosh Kesiraju, and Pavel Smrz. Rethinking the objectives of extractive question answering. *ArXiv*, abs/2008.12804, 2021.

Adam Fisch, Alon Talmor, Robin Jia, Minjoon Seo, Eunsol Choi, and Danqi Chen. MRQA 2019 shared task: Evaluating generalization in reading comprehension. In *Proceedings of the 2nd Workshop on Machine Reading for Question Answering*, pages 1–13, Hong Kong, China, November 2019. Association for Computational Linguistics. doi: 10.18653/v1/D19-5801. URL `https://aclanthology.org/D19-5801`.

Pengcheng He, Xiaodong Liu, Jianfeng Gao, and Weizhu Chen. Deberta: Decoding-enhanced bert with disentangled attention. *ArXiv*, abs/2006.03654, 2021.

Annie Hu, Cindy Wang, and Brandon Yang. Question answering using match-lstm and answer pointer. 2017.

Minghao Hu, Yuxing Peng, Zhen Huang, Xipeng Qiu, Furu Wei, and M. Zhou. Reinforced mnemonic reader for machine reading comprehension. In *IJCAI*, 2018.

Hsin-Yuan Huang, Chenguang Zhu, Yelong Shen, and Weizhu Chen. Fusionnet: Fusing via fully-aware attention with application to machine comprehension. *ArXiv*, abs/1711.07341, 2018.

Mandar Joshi, Eunsol Choi, Daniel S. Weld, and Luke Zettlemoyer. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension, 2017. URL `https://arxiv.org/abs/1705.03551`.

Mandar Joshi, Danqi Chen, Yinhan Liu, Daniel S. Weld, Luke Zettlemoyer, and Omer Levy. SpanBERT: Improving Pre-training by Representing and Predicting Spans. *Transactions of the Association for Computational Linguistics*, 8:64–77, 01 2020. ISSN 2307-387X. doi: 10.1162/tacl_a_00300. URL https://doi.org/10.1162/tacl_a_00300.

Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. *CoRR*, abs/1412.6980, 2015.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. Natural Questions: A Benchmark for Question Answering Research. *Transactions of the Association for Computational Linguistics*, 7:453–466, 08 2019. ISSN 2307-387X. doi: 10.1162/tacl_a_00276. URL https://doi.org/10.1162/tacl_a_00276.

Haonan Li, Martin Tomko, Maria Vasardani, and Timothy Baldwin. MultiSpanQA: A dataset for multi-span question answering. In *Proceedings of the 2022 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 1250–1260, Seattle, United States, July 2022. Association for Computational Linguistics. doi: 10.18653/v1/2022.naacl-main.90. URL https://aclanthology.org/2022.naacl-main.90.

Xiang Lisa Li and Percy Liang. Prefix-tuning: Optimizing continuous prompts for generation. *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, abs/2101.00190, 2021.

Xiaodong Liu, Yelong Shen, Kevin Duh, and Jianfeng Gao. Stochastic answer networks for machine reading comprehension. In *ACL*, 2018.

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

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, Alban Desmaison, Andreas Köpf, Edward Yang, Zach DeVito, Martin Raison, Alykhan Tejani, Sasank Chilamkurthy, Benoit Steiner, Lu Fang, Junjie Bai, and Soumith Chintala. Pytorch: An imperative style, high-performance deep learning library. In *NeurIPS*, 2019.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. Squad: 100,000+ questions for machine comprehension of text, 2016. URL https://arxiv.org/abs/1606.05250.

Ori Ram, Yuval Kirstain, Jonathan Berant, Amir Globerson, and Omer Levy. Few-shot question answering by pretraining span selection. 2021. doi: 10.48550/ARXIV.2101.00438. URL https://arxiv.org/abs/2101.00438.

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. Bidirectional attention flow for machine comprehension. *CoRR*, abs/1611.01603, 2016. URL http://arxiv.org/abs/1611.01603.

Yelong Shen, Po-Sen Huang, Jianfeng Gao, and Weizhu Chen. Reasonet: Learning to stop reading in machine comprehension. *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 2016.

Adam Trischler, Tong Wang, Xingdi Yuan, Justin Harris, Alessandro Sordoni, Philip Bachman, and Kaheer Suleman. Newsqa: A machine comprehension dataset, 2016. URL https://arxiv.org/abs/1611.09830.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and M. Zhou. Gated self-matching networks for reading comprehension and question answering. In *ACL*, 2017.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, and Jamie Brew. Transformers: State-of-the-art natural language processing. In *EMNLP*, 2020.

Ikuya Yamada, Akari Asai, Hiroyuki Shindo, Hideaki Takeda, and Yuji Matsumoto. Luke: Deep contextualized entity representations with entity-aware self-attention. In *EMNLP*, 2020.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W. Cohen, Ruslan Salakhutdinov, and Christopher D. Manning. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *CoRR*, abs/1809.09600, 2018. URL `http://arxiv.org/abs/1809.09600`.

Zhilin Yang, Zihang Dai, Yiming Yang, Jaime G. Carbonell, Ruslan Salakhutdinov, and Quoc V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *NeurIPS*, 2019.

Michihiro Yasunaga, Jure Leskovec, and Percy Liang. Linkbert: Pretraining language models with document links. *ArXiv*, abs/2203.15827, 2022.

Adams Wei Yu, David Dohan, Minh-Thang Luong, Rui Zhao, Kai Chen, Mohammad Norouzi, and Quoc V. Le. Qanet: Combining local convolution with global self-attention for reading comprehension, 2018. URL `https://arxiv.org/abs/1804.09541`.

Urchade Zaratiana, Niama El Khbir, Dennis N'unez, Pierre Holat, Nadi Tomeh, and Thierry Charnois. Dyrex: Dynamic query representation for extractive question answering. 2022.