frontiers | Frontiers in Physics

TYPE Original Research
PUBLISHED 22 December 2022
DOI 10.3389/fphy.2022.1085102

OPEN ACCESS

EDITED BY
Jiang Zhu,
Netskope Inc., United States

REVIEWED BY
Junsong Fu,
Beijing University of Posts and Telecommunications (BUPT), China
Yanping Fu,
Capital University of Economics and Business, China
Wenyu Zhang,
University of Science and Technology Beijing, China

*CORRESPONDENCE
Bo Shen,
bshen@bjtu.edu.cn

SPECIALTY SECTION
This article was submitted to Social Physics, a section of the journal Frontiers in Physics

RECEIVED 31 October 2022
ACCEPTED 02 December 2022
PUBLISHED 22 December 2022

CITATION
Zhang Y, Shen B and Cao X (2022), Learn a prior question-aware feature for machine reading comprehension. *Front. Phys.* 10:1085102. doi: 10.3389/fphy.2022.1085102

Learn a prior question-aware feature for machine reading comprehension

Yu Zhang[1], Bo Shen[2]* and Xing Cao[1]

[1]School of Electronic and Information, Beijing Jiaotong University, Beijing, China, [2]Key Laboratory of Communication and Information Systems, Beijing Municipal Commission of Education, Beijing Jiaotong University, Beijing, China

Machine reading comprehension aims to train machines to comprehend a given context and then answer a series of questions according to their understanding of the context. It is the cornerstone of conversational reading comprehension and question answering tasks. Recently, researches of Machine reading comprehension have experienced considerable development with more and more semantic features being incorporated into end-to-end neural network models, such as pre-trained word embedding features, syntactic features, context and question interaction features, and so on. However, these methods neglect the understanding of the question itself and the information sought by the question. In this paper, we design an auxiliary question-and-answer matching task to learn the features of different types of questions and then integrate these learned features into a classical Machine reading comprehension model architecture to improve its ability to comprehend the questions. Our auxiliary task relies on a simple Question-Answer Pairs dataset generated by ourselves. And we incorporate the learned question-type information into the Machine reading comprehension model by prior attention mechanism. The model we proposed is named PrA-MRC (**Pr**ior **A**ttention on Machine reading comprehension). Empirical results show that our approach is effective and interpretable. Our Question-Answer Pairs model achieves an accuracy of 84% and our PrA-MRC model outperforms the baseline model by +0.7 EM and +1.1 F1 on the SQuAD dataset.

KEYWORDS

question and answer pairs, prior attention mechanism, transfer learning, machine reading comprehension (MRC), nature language processing (NLP)

1 Introduction

Natural language understanding is a significant but also challenging direction of natural language processing (NLP), which refers to making machines be capable of understanding the semantic of natural language as humans do. To achieve this, researchers imitate process of language learning of humans and design a variety of unique tasks to train neural networks in the expectation that this model can learn specific information. Machine reading comprehension (MRC) is one of these tasks that mimic the

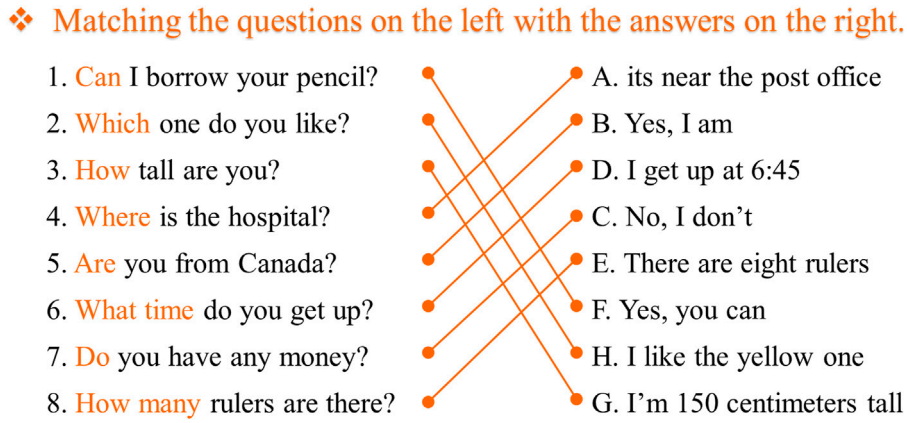

FIGURE 1
An example of questions and answers linking problem in students' examination.

humans leaning process. Since humans leverage various forms of questions to assess how well language learners have mastered the language, MRC also generates various types of tasks to evaluate the machine's understanding level of the article [1], such as reading comprehension with multiple choice [2], cloze test [3], span extraction [4, 5], free answering [6], *etc.* Although the form of the answer varies, any type of MRC model must fully understand the question to find a more appropriate answer. However, the existing MRC models represent questions in the same way as representing background context which cannot enhance the MRC model's ability to comprehend the questions. In this paper, we focus on the information naturally contained in questions for seeking answers. And we are inspired by questions and answers linking problem in students' examination. Questions and answers linking problem is to match questions and answers without any background context, as shown in Figure 1. Different kinds of questions seek for different information, so we can infer the answer type by the interrogative pronoun in question without context. We believe that question and answer pairs contain rich information that can guide questions to find answers in the passage, so we will mine this information for guiding span-based machine reading comprehension tasks. Span-based MRC tasks train the model to identify the suitable answer span from the given passage, which has received growing interest these years since its extractive answer balances the understanding and evaluability of the model.

Many existing MRC models take advantage of external knowledge to enrich semantic features, especially models based on flourishing pre-trained language models such as ELMo [7], GPT [8], Bert [9] are equivalent to incorporating global semantic knowledge to represent each word. The MRC models using character embedding can be regarded as incorporating features of uncommon words. And the MRC models based on bidirectional attention integrate the interactive features between context and question. Some other MRC models incorporate features of common sense relying on a manually annotated knowledge base. However, there are few models considering the characteristics of different types of questions in the reading comprehension task. It is obvious that interrogative pronoun in a question indicates the type of information to be sought in this question. Zhang et al. [10] explicitly encoded the type of the first word in a question to an 11-dimensional one-hot vector but the model only considers the type of question itself and remains unaware of the type of information sought by the question. Tayyar et al. [11] created an entity identification method to extract entities belonging to different classes of questions in their manual taxonomy, which required pre-defining all the question type. In this article, we learn the information and answer way of different question types in a simple and automatic method. And then we integrate this features in MRC model to enhance its question understanding ability.

We regard the information of different kinds of questions seek as a general question embedding model, which is inspired by [12]. Conneau et al. trained universal sentence representations on the supervised data of Stanford Natural Language Inference dataset (SNLI) [13]. They hypothesize natural language inference (NLI) task is a high-level understanding task that involves reasoning about the semantic relationships between premise and hypothesis. The experiment shows their sentence embeddings reach the best results in the transfer tasks. In this paper, we focus on questions and answers in machine reading comprehension tasks, so we train a question embedding which involves the information sought by this question for MRC models. Specifically, we generate a simple **Q**uestion-**A**nswer **Pairs** (**QApairs**) dataset to train the question representations instead of universal sentence representations trained on SNLI

dataset. After that, we enroll the learned question representations containing rich information into the baseline MRC model through the prior attention mechanism [14]. Empirical results show that our **PrA-MRC** (**Pr**ior **A**ttention on **MRC**) model achieves competitive performance in span-based machine reading comprehension task and visualized analysis show that our separate question representations involve additional question type information indeed.

To sum up, the main contributions of this paper are as follows:

1) We propose a new simple task QApairs to learn the implicit interaction features between all kinds of questions and answers for subsequent MRC task.
2) We encode questions in machine reading comprehension task by fixed sentence encoder which is trained on the QApairs dataset. And we use the question embedding as prior question-aware features to train the baseline MRC model.
3) We demonstrate the effectiveness of our method through a series of experiments. And we analyze the experimental results in detail for future research.

This article is organized as follows. We first expound the related works in machine reading comprehension and transfer learning which are closed to this research, then we describe the proposed dataset QApairs and the sentence encoder based on the QApairs. Subsequently, we present the architecture of PrA-MRC model and the basic design choices of the model. Finally, we visualize and analyze our results in SQuAD dataset and draw some conclusions for future work.

2 Related works

2.1 MRC models

The neural MRC models have flourished in recent years, which is mainly benefit from the improvement in two aspects. On the one hand, the rise of computing power led to data-driven models and large pre-trained language models. Specifically, Devlin et al. [9], proposed the pre-trained model BERT and it obtain remarkable results on 11 natural language processing tasks including MRC task. Yang er al. [15] improved XLnet to overcome the limitations of BERT by its autoregressive formulation and make a further progress. All those pre-trained models demonstrate the benefits of large amounts of data. However, the disadvantage of these models requiring more computing resources and time is also obvious. Moreover, BERT-based models connect the question and context using 'SEP' and then inputs them together, and there is no deeper interaction between question and context except the global semantic information contained by each word. On the other hand, the expansion of semantic and syntactic features enhanced MRC models' performance by well-designed neural models. Seo et al. [16] fused the semantic interactive information of question and context through a two-way attention mechanism, and this paper also laid a structural framework for inputting question and context. Huang et al. [17] proposed a fully-aware multi-level attention mechanism to capture the entire information from the lowest word-level embedding up to the highest semantic-level representation [18]. Proposed using syntactic dependency parse tree to obtain better representations of context words by incorporating explicit syntactic constraints into attention mechanism. In this paper, we integrate prior question-aware semantic information learned without context into the MRC model to strengthen its understanding ability.

2.2 Multi-turn MRC models

Multi-turn machine reading comprehension is also referred to as conversational machine reading comprehension, which can be seen as incorporating historical information features into the MRC model. In this article, we use two multi-turn machine reading comprehension datasets to train our sentence encoder. For multi-turn MRC, the simplest end-to-end neural model is to add the historical dialogue turns to current question and feed them to the original MRC model. Others integrate the historical semantic information or historical answers' position information. For example, Huang et al. [19], defined Flow operation that uses entire hidden representations generated by intermediate process when answering previous question to feeds the MRC model. The specific operation is to concatenate the representation of each original context word and the representation of each context word in the inference stage. Yeh et al. [20] extended the conception of Flow and proposed an approach named Flow-Delta to model information gain in Flow by subtracting two representations. Qu et al. [21] appended a history answer embedding which denotes whether a context token is a part of history answers or not to the three embeddings of BERT. And Qu et al. [22] expanded on the previous work with a history attention module to choose whether to consider a question turn. All those models integrate historical information on the basic framework of MRC models, and most of models uses concatenate of vector or attention mechanism to fuse the historical information.

2.3 Transfer learning

Transfer learning [23] is designed to improve the performance of target task by transferring the knowledge learned from related source tasks. This approach can reduce the dependence of target task on large amounts of data, and it also benefit task domain applicability, so it has been successfully

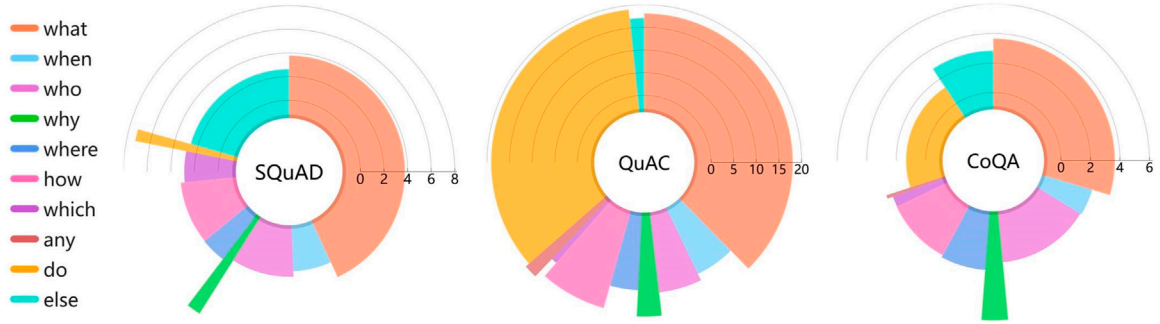

FIGURE 2
A visual analysis of SQuAD, QuAC and CoQA datasets. Each sector in a rose chart represents the proportion of one kind of questions, and the radius represents the average length of the answers to such kind of questions.

applied to plentiful tasks in deep learning. The most classic example is that features trained on ImageNet [24] can be transferred to many other computer vision tasks such as face recognition [25] and visual question answering [26]. There are numerous attempts to apply transfer leaning in machine reading comprehension and some achievements have been made. Jiang et al. [27] improved multi-choice reading comprehension by transferring the experience of single-choice decision. Conneau et al. [12] trained universal sentence representations and transefer it to a wide range of NLP tasks including but not limited to machine reading comprehension. Kundu S et al. [28] learned to identify follow-up questions through an auxiliary task and the simple dataset they generated for multi-turn machine reading comprehension. Inspired by all this applications of transfer leaning, and considering the question answer pairs inherently informative, we genarate a simple question and answer pairs dataset and transfer the learned knowledge on this dataset to MRC models.

3 QA pairs dataset

This section we are going to analyze the source datasets used to prepare our QApairs dataset and describe the process of preparing it, and then we will introduce how to use the QApairs dataset to train a sentence encoder that can capture the implicit information sought by different types of questions.

3.1 Question aware dataset analysis

QApairs dataset relies on three source datasets, namely SQuAD [4], QuAC [29], CoQA [6]. SQuAD is the first large scale span-based MRC dataset, and it can be regard as a milestone of machine reading comprehension task. SQuAD dataset contains 536 articles and more than 100 K questions from WiKipedia. For each article, the crowd-workers should ask five questions and mark the text spans in given passage as answers. QuAC is a multi-turn question answering dataset which contains 100 K questions. This dataset is generated by a student asking some open-ended questions for seeking more information based on the section's title or first paragraph only and a teacher answering the questions based on the full section. Similar to QuAC, CoQA is a conversational question answering dataset, which is generated by two annotators having conversation about a passage, both questioner and answerer can see the full context and history conversation that happened until now. But unlike QuAC, the answers in CoQA dataset are free-form texts.

We perform a visual comparison of these three datasets as shown in Figure 2. The Rose chart represents the distribution of first word in questions, and radius of each slice represents the average length of answers to questions in that category. It is obvious to see that nearly half of questions in SQuAD and QuAC are dominated by *what* questions. And the distribution of question type in CoQA is more uniform. We can also see the intuitive phenomenon that the answers to *why* questions are distinct longer because *why* answers are usually explanatory. In addition, the answers of QuAC are distributed between 15–20 words, while the answers of the other two datasets are distributed between two to eight words.

All types of questions in three datasets are shown in Table 1. For the key word *do* also includes its past tense *did* and third person *does*, so does words *is, have*. All the question-answer examples are selected from the above three QA datasets. We also found *else* questions have key words, however, the key words are put at the end of the questions. Different types of questions need to be answered in different ways according to the information sought by those questions. To be specific, when the interrogative pronoun of a question is *when*, the answer to this question should be time related words (except cannot answer question). It should

TABLE 1 Different kinds of examples in SQuAD, QuAC and CoQA datasets.

Question type	Key words	Example
Description	What	Q: What movie did Beyonce act in 2006?
		A: The Pink Panther
Time	When	Q: When did Beyonce begin her second world tour?
		A: March 2009
Person	Who, Whom, Whose	Q: Who beat out Beyonce for Best Female Video?
		A: Taylor Swift
Reason	Why	Q: Why does genocide often go unpunished?
		A: genocide is more often than not committed by the officials in power
Place	Where	Q: Where is Topshop located?
		A: London
Way	How	Q: How much more that the budget did the film gross?
		A: 60 million
Choice	Which	Q: Which singer did Beyonce portray in Cadillac Records?
		A: Etta James
Supplement	Any, Anything, Anyone, Anymore	Q: Anything else you found interesting?
		A: referred to Monroe as the "father" of bluegrass
General	Do, Is, Have, Can, Could, Should, Would	Q: Does he choose one?
		A: They select Louise de la Valliere for this part
		Q: Did she live alone? (CoQA)
		A: no
Else	After, In, Besides	Q: Ag3Cu is one intermetallic that tin forms; what's the other one?
		A: Cu5Sn6

be noted that a general question is usually answered with either *yes* or *no*, which is indeed in CoQA. But in SQuAD and QuAC, since the answer is a span selected from a paragraph, the answer to a general question is a piece of reasoning text which supports *yes* or *no* answer.

We aim to learn the information contained in plenty of question-answer pairs without any background context. Our motivation is that a question semantic embedding and its matching answer semantic embedding obtained by our sentence encoder are closer in some semantic vector space. Thus, our model receives a sentence pair and estimate whether the sentence pair matches, which is a classification task.

3.2 Generate QApairs

We introduce the generation process of QApairs dataset as follows.

1) First, we extract question and answer pairs from three source datasets as examples. Each question-answer pair is numbered so that matching question and answer have the same ID.
2) Whether or not a question is answerable depends on the relevant passage. Since our goal is to learn representations through the question-answer pairs only, we remove examples where the answer is 'Can not answer'.
3) All question-answer pairs are added as positive examples and labeled with 'match'.
4) Randomly select question and answer separately, and if their IDs are different, the question-answer pair is added as a negative example and labeled with 'not match'. The ratio of positive examples and negative examples is kept in 1:1.
5) Shuffle all the question-answers pairs.
6) Divide all the examples into training set and validation set with a ratio of 8:2.

Notice that the pairs with question type as *else* are not removed. On the one hand, there are still keywords in such question sentences,

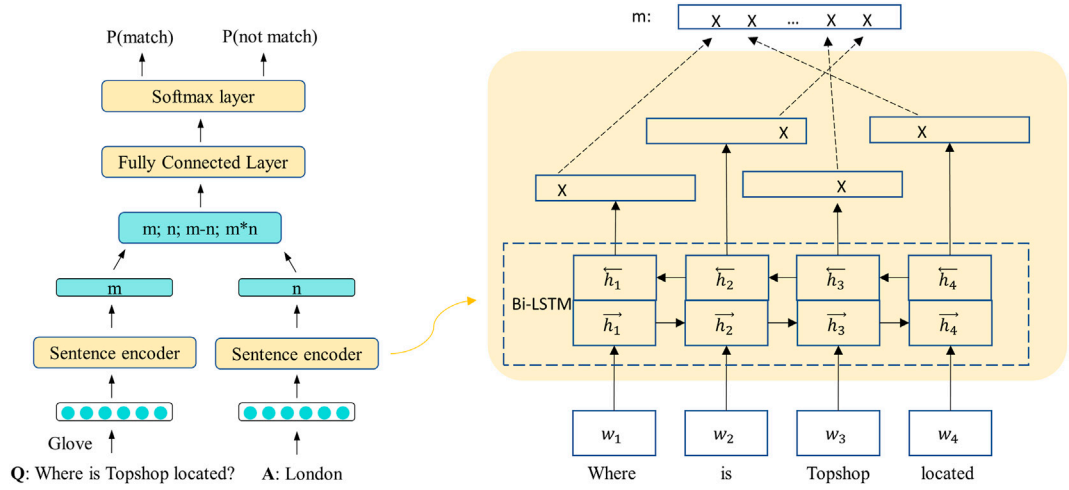

FIGURE 3
QApairs classification model architecture. On the left is the overall structure of the QApairs classification model and on the right is the detail of the sentence encoder.

which does not affect the implicit information we aim to learn. On the other hand, we expect the learned model to be more robust by enhancing confusion of the dataset.

Moreover, we keep four QApairs datasets from different sources: QApairs-I from CoQA only, which consists of 233 k sentence pairs, QApairs-II from QuAC only, which consists of 190 k sentence pairs, QApairs-III from SQuAD only, which consists of 244 k sentence pairs and QApairs-IV from all three dataset, which consists of 668 k sentence pairs.

3.3 QApairs classification model

QApairs classification model is a simple binary classification model which inputs the question-answer pairs and determines whether the question-answer pairs match. Considering that the information we are going to learn about the question would be applied to subsequent machine reading comprehension task, we chose to encode the two sentences separately. Our QApairs model is a typical architecture of generic sentence encoder [12] which uses a shared sentence encoder, as illustrated in Figure 3. First, we conduct word embedding for each word in sentence pairs. $\boldsymbol{Q}$ and $\boldsymbol{A}$ represent the input sentences respectively. Second, we encode the sentence pairs using a shared sentence encoder to generate sentence representations m and n. Then we fusion these two representations by concatenating m and n, the absolute element-wise difference of m and n and the element-wise product of m and n, which can be describe as $[m; n; m - n; m^*n]$. The resulting vector, which captured the joint sematic information from both question and answer, is fed into a fully connection layer to extract features and finally to a SoftMax layer for binary classification.

3.4 Sentence encoder

Sentence encoder, which can also be regarded as feature extraction, is used to encode each token in a sentence into a fixed size vector representation with contextual information. At present, common neural network architectures for sentence encoding can be classified into three categories: Recurrent Neural Networks (RNN), Convolutional Neural Networks (CNN) and Transformer. RNNs such as long short-term memory (LSTM) and Gate Recurrent Unit (GRU) networks can capture context dependencies of different lengths of sentences. Convolutional Neural Networks runs faster due to its parallelism. Transformer combines both effective and efficient cause it based solely on attention mechanisms. Here, we choose Bi-directional LSTMs for our QApairs classification model cause the input sentences are not very long and it is transferable for subsequent MRC model.

The input of our sentence encoder is two sentences, question $\boldsymbol{Q} = \{q_1, q_2, \ldots q_J\}$ and answer $\boldsymbol{A} = \{a_1, a_2, \ldots a_E\}$ that have J and E words respectively. The output are the semantic representations of the two sentences. We obtain a set of vectors $\{ h_t = [\overrightarrow{h_t}; \overleftarrow{h_t}] \}$, which is the concatenation of the output of a forward LSTM and a backward LSTM for each word in the input sentences. And then we combine the varying numbers of h_t to compose a fixed-size representation for each sentence by max

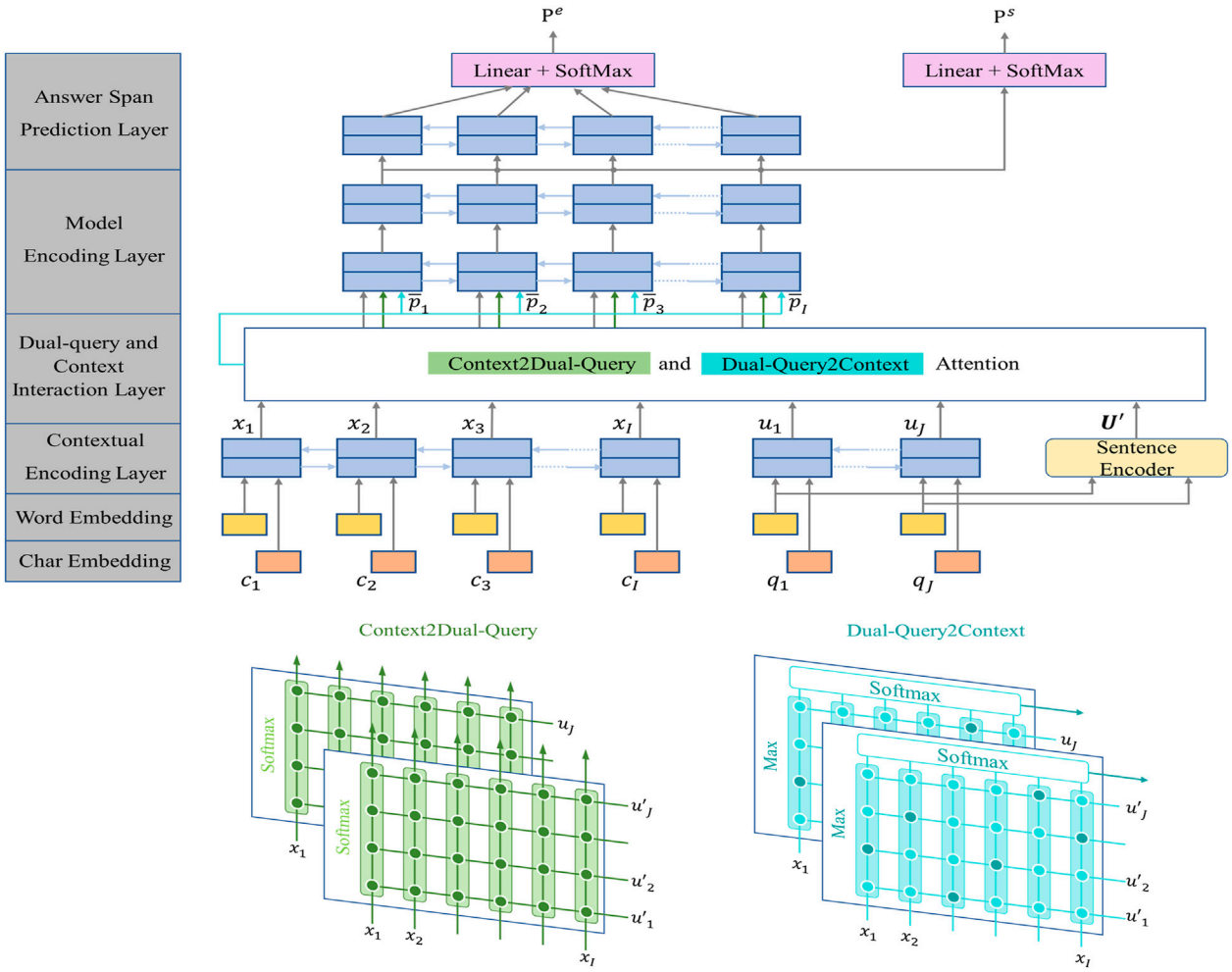

FIGURE 4
PrA-MRC model architecture. The bottom half of the picture is the detail attention between context and Dual-query (basal query and type-information-aware query).

pooling operation, which selecting the maximum value from each dimension of the hidden units. Finally, we obtain the sentence representation m, $m \in \mathbb{R}^{2d}$ for question $\boldsymbol{Q}$ and representation n, $n \in \mathbb{R}^{2d}$ for answer $\boldsymbol{A}$ by the shared sentence encoder.

4 PrA-MRC: Prior attention to MRC

After training individual QApairs model, we return to our primary MRC task with the learned implicit features by QApairs. The goal of span-based MRC task is to understand a given question and context, looking for a contiguous text span within the given context as an answer. We introduce a **PrA-MRC** model which fuses the learned prior attention into the classical MRC model **Bi-DAF** [16]. Our model consists of five fundamental blocks: embedding layer, contextual coding layer, dual-query and context interaction layer, modeling layer and output layer. The overall PrA-MRC model is illustrated in Figure 4.

4.1 Embedding layer

Embedding layer, including word embedding and character embedding, maps each word in sentences to a high-dimensional vector space. Following [30], we conduct character embedding using Convolutional Neural Networks (CNN) and word embedding using pre-trained word vectors Glove [31]. Let $\boldsymbol{C} = \{c_1, c_2, \ldots c_I\}$ and $\boldsymbol{Q} = \{q_1, q_2, \ldots q_J\}$ (same with $\boldsymbol{Q}$ in QApairs

model) represent the sentences of the input context and query, respectively. The vectors obtained from character embedding and word embedding are concatenated and passed through a Highway Network [32] to get the final vector representation of each sentence word. Hence, the output of embedding layer is two matrices $\boldsymbol{C} \in \mathbb{R}^{d\times I}$ and $\boldsymbol{Q} \in \mathbb{R}^{d\times J}$, where d is the dimension of each word vector representation, and I and J are the length of context sentence and query sentence respectively.

4.2 Contextual encoding layer

Contextual encoding layer is re-encoding vectors of words to make them infuse contextual information. A Bidirectional Long Short-Term Memory Networks (Bi-LSTMs) is added to the top of the embedding layer for context awareness. Then we obtain two matrixes: $\boldsymbol{X} \in \mathbb{R}^{2d\times I}$ for the context word embeddings $\boldsymbol{C}$ and $\boldsymbol{U} \in \mathbb{R}^{2d\times J}$ for the query word embeddings $\boldsymbol{Q}$, where I and J are still the length of sentences in input context and query respectively. While the dimension of each word becomes $2d$ because we concatenate the output of the forward LSTM and the backward LSTM, each with dimension d, to obtain the context awareness in both directions.

Besides, we encoder the question $\boldsymbol{Q}$ by the fixed sentence encoder trained by QApairs model. This sentence encoder has same structure and hidden units with contextual encoding layer, so we get the type-information-aware query representation $\boldsymbol{U}' \in \mathbb{R}^{2d\times J}$.

4.3 Dual-query and context interaction layer

The interaction layer in original BiDAF links and fuses information between the context and query and thus generates a series of query-aware feature vectors for every word in given context. On this basis, we carry out interactive calculation between context and the query with prior type information produced by QApairs model and generates a series of query-type-information-aware feature vectors for every word in given context either. Then, we aggregate query-aware feature vectors and query-type-information-aware feature vectors for each context word.

The inputs of this layer are the context representation $\boldsymbol{X}$, the basal query representation $\boldsymbol{U}$ and the type-information-aware query representation $\boldsymbol{U}'$, ($\boldsymbol{X} \in \mathbb{R}^{2d\times I}$, $\boldsymbol{U}, \boldsymbol{U}' \in \mathbb{R}^{2d\times J}$). We compute the interaction among these vectors by attention mechanism: from context to both queries as well as from both queries to context. And those attentions are derived from two similarity matrices that are calculated in the same way. The similarity matrices S, S' between the context representation $\boldsymbol{X}$ and the dual query representation $\boldsymbol{U}$, $\boldsymbol{U}'$ is computed by

$$S_{ij} = \partial\left(X_{:\,i}, U_{:\,j}\right) \in \mathbb{R} \tag{1}$$

$$\partial(x,\ u) = linear(x) + linear(u) + linear(x \circ u) \tag{2}$$

$$S'_{ij} = \partial\left(X_{:\,i}, U'_{:j}\right) \in \mathbb{R} \tag{3}$$

$$\partial(x, u') = linear(x) + linear(u') + linear(x \circ u') \tag{4}$$

where $X_{:\,i}$ is i-th column vector of context representation $\boldsymbol{X}$ and can be regard as i-th word in context. $U_{:\,j}$ is j-th column vector of query representation $\boldsymbol{U}$ and can be regard as j-th word in query. So S_{ij} indicates the similarity between i-th word in context and j-th word in query. *linear* () is a linear layer that maps a $2d$ vector to one dimension. $\partial(x,\ u)$ is the similarity calculation method which is the sum of context word representation mapping, query word representation mapping and elementwise multiplication between these two representations. Another similarity matrix between context representation $\boldsymbol{X}$ and query with type information representation $\boldsymbol{U}'$ is computed in the same way, which is described by formula (3) and (4). Then we use $\boldsymbol{S}$ and $\boldsymbol{S}'$ ($\boldsymbol{S}$, $\boldsymbol{S}'_{ij} \in \mathbb{R}^{I\times J}$) to obtain query-aware context representations and query-type-information-aware context representations.

4.3.1 Query-aware context representation

Query-aware context representation is generated by two directions attention, context-to-query attention as well as query-to-context attention.

Context-to-query attention represents the importance of each query word to context. We obtain query attention weight α_i for i-th context word by normalizing i-th column in similarity matrix $\boldsymbol{S}$ (Formula 5), subsequently the i-th context word representation that fused query, $\tilde{U}_i$, is the product of α_i and the query representation $\boldsymbol{U}$ (Formula 6). Hence, we obtain $\tilde{\boldsymbol{U}} \in \mathbb{R}^{2d\times I}$ for all context words.

$$\alpha_i = softmax(\boldsymbol{S}_{i:}) \in \mathbb{R}^J \quad , \sum_{j=0}^{j=J} \alpha_{ij} = 1 \tag{5}$$

$$\tilde{U}_i = U\alpha_i \in \mathbb{R}^{2d} \tag{6}$$

Query-to-context attention expresses which context words are most relevant to each query word. It is important because the most relevant words are likely to be the answer words. We perform maximum function across the column of $\boldsymbol{S}$ and execute softmax on its result to obtain the context attention weight b (Formula 7). Then the attended context vector $\tilde{x}_i$ can be calculated by Formula 8. The vector $\tilde{x}_i$ indicates the weighted sum of the most relevant words in the context with respect to the query. Finally, we repeat $\tilde{x}_i$ for I times to get $\tilde{\boldsymbol{X}} \in \mathbb{R}^{2d\times I}$ for all context words.

$$b = softmax(max_{col}(\boldsymbol{S})) \in \mathbb{R}^I \tag{7}$$

$$\tilde{x}_i = \boldsymbol{X}b \in \mathbb{R}^{2d} \tag{8}$$

Now we obtain two attended context representations in same shape $\mathbb{R}^{2d\times I}$, and they are combined with original contextual

embeddings to yield query-aware representations $\boldsymbol{P} \in \mathbb{R}^{8d\times I}$. We define $P_{:i}$ by

$$P_{:i} = \left[X_{:i};\ \tilde{U}_{:i};\ X_{:i} \circ \tilde{X}_{:i};\ X_{:i} \circ \tilde{U}_{:i}\right] \in \mathbb{R}^{8d} \tag{9}$$

where $P_{:i}$ is the i-th column vector which is corresponding to each word in the context. The symbol ; represents vector concatenation, and $\circ$ represents elementwise multiplication.

4.3.2 Query-type-information-aware context representation

Query-type-information-aware context representation is computed in the same way as query-aware context representation except the query with type information representation $\boldsymbol{U}'$ is used instead of that original query representation $\boldsymbol{U}$. Query-type-information-aware context representation $\boldsymbol{P}' \in \mathbb{R}^{8d\times I}$ is computed as follows.

$$\alpha_i^{'} = softmax\left(S_{i:}^{'}\right) \in \mathbb{R}^{J}, \sum_{j=0}^{j=J} \alpha_{ij}^{'} = 1 \tag{10}$$

$$\tilde{U}_i^{'} = \boldsymbol{U}^{'} \alpha_i^{'} \in \mathbb{R}^{2d} \tag{11}$$

$$b' = softmax\left(max_{col}\left(\boldsymbol{S}'\right)\right) \in \mathbb{R}^{I} \tag{12}$$

$$\tilde{x}_i' = \boldsymbol{X} b' \in \mathbb{R}^{2d} \tag{13}$$

$$P'_{:i} = \left[X_{:i};\ \tilde{U}'_{:i};\ X_{:i} \circ \tilde{X}'_{:i};\ X_{:i} \circ \tilde{U}'_{:i}\right] \in \mathbb{R}^{8d} \tag{14}$$

Considering that we have two context representations now, one is query-aware context representation $\boldsymbol{P} = \{p_1, p_2, \cdots, p_I\}$, the other is query-type-information-aware context representation $\boldsymbol{P}' = \left\{p_1^{'}, p_2^{'}, \cdots, p_I^{'}\right\}$. Formally, the final output representation $\bar{\boldsymbol{P}} = \{\bar{p}_1, \bar{p}_2, \cdots, \bar{p}_I\}$ of the interaction layer is computed by a hyper-parameter $\gamma \in (0, 1)$ as follow.

$$\bar{p}_i = \gamma p_i + (1-\gamma) p_i^{'} \tag{15}$$

4.4 Model encoding layer

Model encoding layer is similar to the contextual encoding layer in that it adopts bi-directional LSTMs to capture the interactions among context representations. However, the input $\bar{\boldsymbol{P}}$ of the model encoding layer already encoded the query-aware and query-type-information-aware representations in context words, which is different from independent input $\boldsymbol{X}$ and $\boldsymbol{U}$ of contextual encoding layer. We use two layers of Bi-LSTMs to encode the dual-query context representations, each direction of LSTM with dimension d. Therefore, we obtain the output matrix $\boldsymbol{M} \in \mathbb{R}^{2d\times I}$. The output of this layer is expected to contain all the information for answering the current question.

TABLE 2 Comparison in performance of four QApairs datasets.

Dataset	Description	Q/A pairs	Acc
QApairs-I	CoQA only	233 k	0.78
QApairs-II	QuAC only	190 k	0.71
QApairs-III	SQuAD only	244 k	0.69
QApairs-IV	All	668 k	0.84

4.5 Answer span prediction layer

Since our task is to select the starting and ending position of the answer span in given context. We can regard answer span prediction as a soft classification task, predicting the probability of each context word being the starting word or ending word. Hence, it is appropriate to use softmax to generate probabilities across all context words. We follow the method of BiDAF to construct this layer. The probability distribution of the start positions P^s over all context words calculated by

$$P^s = softmax\left(linear\left(\left[\bar{\boldsymbol{P}}; \boldsymbol{M}\right]\right)\right) \tag{16}$$

where *linear* () maps a $10d$ ($\bar{\boldsymbol{P}} \in \mathbb{R}^{8d\times I}$ *and* $\boldsymbol{M} \in \mathbb{R}^{2d\times I}$) vector to one dimension. Then we pass $\boldsymbol{M}$ to another Bi-LSTMs module and get $M' \in \mathbb{R}^{2d\times I}$, which, in our opinion, fuses the start position information and can be used for predicting the end position. And then we use $\boldsymbol{M}'$ and $\bar{\boldsymbol{P}}$ to calculate the probability distribution of the end position P^e. The formula is donated by

$$P^e = softmax\left(linear\left(\left[\bar{\boldsymbol{P}}; \boldsymbol{M}'\right]\right)\right) \tag{17}$$

4.6 Loss function

We use the sum of the negative log likelihood of the truly start and end positions by the predicted distributions, and the sum is averaged over all instances. The formula is donated by

$$L(\theta) = -\frac{1}{N}\left(\sum_1^N i^s \log\left(p_i^s\right) + \sum_1^N i^e \log\left(p_i^e\right)\right) \tag{18}$$

where L is training loss, a function of all trainable weights θ in the model. N is the number of instances in dataset. i^s is one when i is the truly start index and i^e is one when e is the truly end index, otherwise they will be 0. The training loss will be minimized.

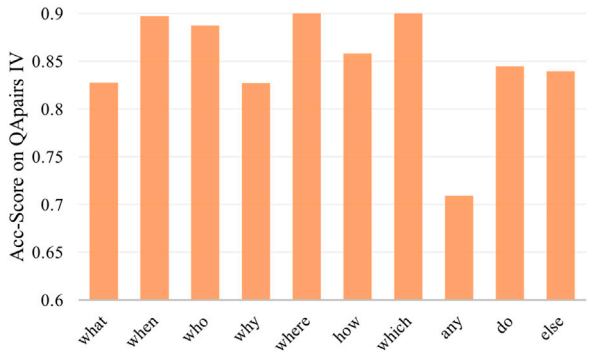

FIGURE 5
QApairs results for different kinds of questions on QApairs IV.

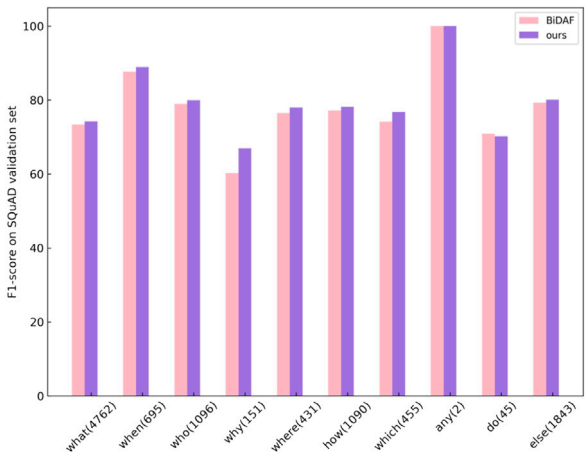

FIGURE 7
Comprision of F1-score between BiDAF and PrA-MRC for different kinds of questions on SQuAD.

5 Experiments and analysis

5.1 Implementation details

Our two models are trained independently. For QApairs model, Adam optimizer is used with a learning rate of 1e-3. A dropout [33] is used for the linear layer and LSTM layers with a rate of 0.2. We use open-source 300-dim Glove [31] vectors pre-trained on Common Crawl 840 B as fixed input word embeddings and all the hidden layers have 100 units which is same as BiDAF for fusing the prior attention information. A minibatch size of 128 for 15 epochs are adapted for training, and it takes approximately 4 h on a single Tesla V100 GPU.

For PrA-MRC model, we keep parameters setting same as the baseline (Bi-DAF) model to prove the effectiveness of information extracted from question-answer pairs. Since the extra task does not perform well on the first three datasets and this may cause incorrect propagation, (see Table 2), we only append QApairs IV dataset for the PrA-MRC model. Word embedding is 300-dim Glove vectors and Char embedding is 100-dim. The final word representation is 200-dim. The hidden size d of LSTM layers is 100. AdaDelta [34] optimizer is used with an initial learning rate of 0.5. A dropout is used for CNN layers, LSTM layers and the linear layer before the final softmax layer with a rate of 0.2. The hyper-parameter γ combined the two context representations is 0.5. A minibatch size of 50 for 15 epochs are adapted on a single Tesla V100 GPU for training, and the training process takes approximately 7 h.

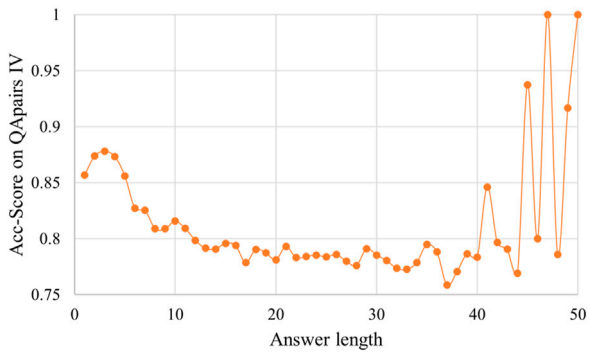

FIGURE 6
QApairs results for different lengths of answers on QApairs IV.

5.2 Results and analysis

Results and analysis will revolve around three questions. 1) How is the performance of auxiliary QApairs task? 2) Whether the auxiliary task is effective when added to reading comprehension task? 3) Whether the auxiliary task learn the prior question-type features we aim to learn?

5.2.1 QApairs study

This section reports performance of our QApairs model. All four datasets are divided into training set and test set, and only the accuracy result *acc* on the test set is reported. *Acc* refers to the proportion of correctly classified QA pairs to all QA pairs, as shown in Table 2. It can be seen from the table that a good result can be obtained by simple Bi-LSTM + Maxpooling structure, and the accuracy becomes better when all three source datasets are appended. It is worth noting that the model performs well on CoQA only dataset, likely because that answers in CoQA dataset are free-formed texts and thus there are greater correlations between questions and answers.

We also analyzed the matching accuracy of different kinds of questions in QApairs-IV dataset as shown in Figure 5, which fits

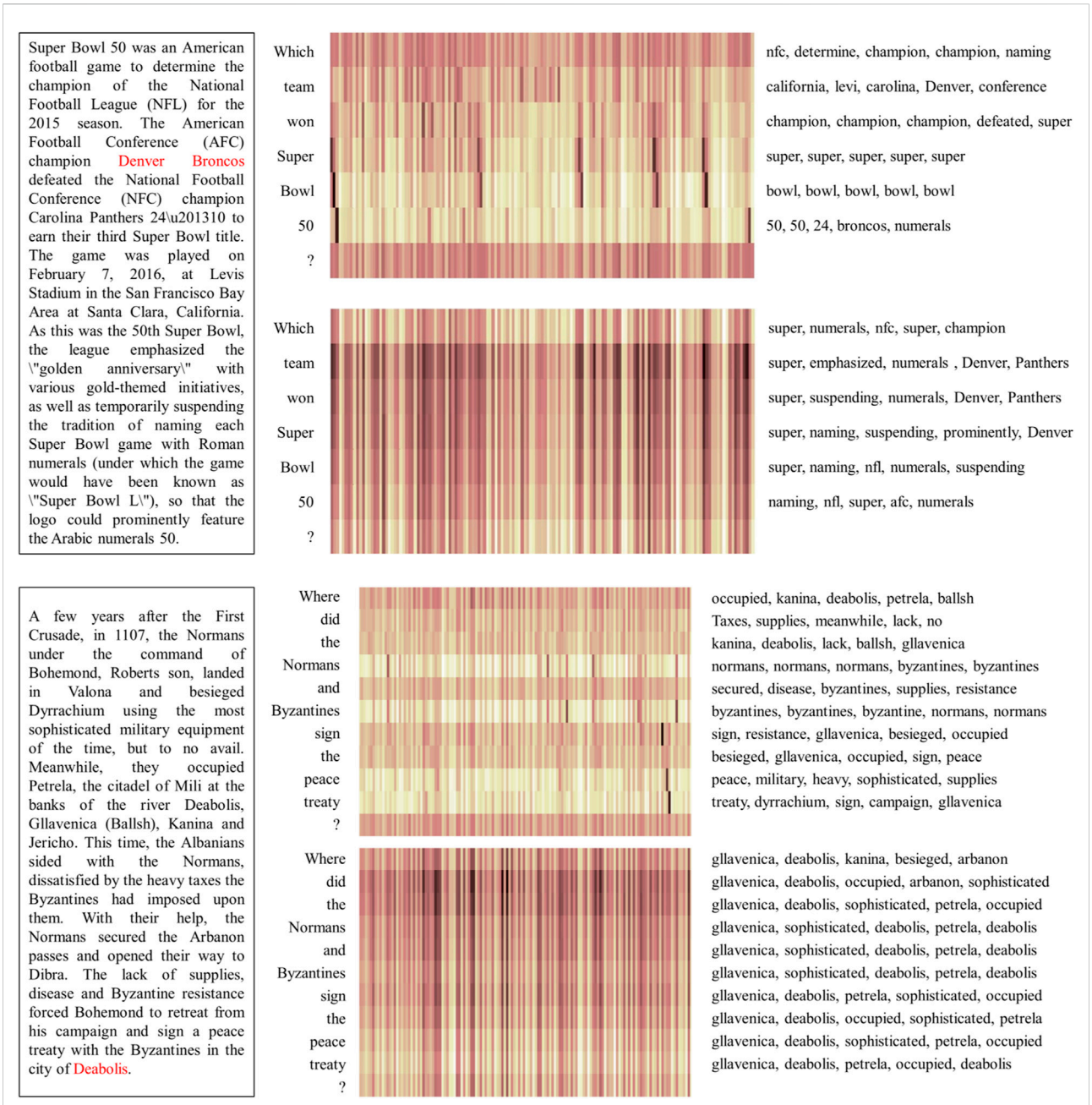

FIGURE 8
Attention matrices for dual-query and context tuples in two visual examples. In each figure, the Attention matrices (each column is a context word, each row is a question word) are shown in the middle palette. The darker the bar, the higher the value of attention. Above the middle palette is the attention matrix of the original question and context, below the middle palette is the attention matrix of the question with prior knowledge and context. The left passage shows the given context with correct answer in red, and the right texts show the top five context words with higher attention points for each question word.

human intuition that the matching accuracy is higher for specific questions (such as *when, where,* and *who*) than for descriptive questions (such as *why*, *any*). The best-performing *which-question* and *where-question* achieve more than 90% accurate. And yet *any-question* performs significantly worse than the others, which we suspect is reasoning required to answer *any-question*. For instance, to answer question *'any other charity works?'* (From QuAC dataset) must exclude the charity work mentioned before.

Another analysis is about the accuracy of different lengths of answers as show in Figure 6. At the beginning, accuracy decreases with the increase of answer length, then accuracy keeps around 0.78 when lengths of answers in range (13, 40). When the answer length is greater than 40, accuracy oscillates

TABLE 3 Comparison in performance of the baseline approaches and our model on SQuAD.

Model	EM	F1
Dynamic Chunk Reader [35]	62.4	70.9
Fine-Grained Gating [36]	62.4	73.3
Match-LSTM with Bi-Ans-Ptr [37]	64.7	73.7
Dynamic Coattention Networks [38]	66.2	75.8
FABIR [39]	67.7	77.6
BiDAF [16]	66.1	76.3
PrA-MRC (with QApairs-I)	66.2	77.1
PrA-MRC (with QApairs-II)	66.6	76.9
PrA-MRC (with QApairs-III)	65.9	76.3
PrA-MRC (with QApairs-IV)	66.8	77.4
BiDAF (with Bert)	75.0	83.6
PrA-MRC (with QApairs-IV and Bert)	76.4	85.1

because there are few instances with the answer length greater than 40. For example, there are only four instances with the answer length of 47 and five instances with the answer length of 50, and they all match correctly.

5.2.2 PrA-MRC study

Table 3 reports our PrA-MRC model performance on SQuAD dataset. Following all MRC tasks, we use answer exactly match (EM) and macro-average F1 score of answer overlap as our evaluation metric. For a relatively fair comparison, we don't adopt the comparison models based on BERT or other large scale pretrained model. And an illustration is that we directly use the results from original paper for five models above, but we use the results of BiDAF model by our re-implementation. It can be found that PrA-MRC is effective especially PrA-MRC with QApairs-all yields substantial improvement over the baseline BiDAF (EM+0.7, F1+1.1), showing the effectiveness of implicit information captured by QApairs. However, PrA-MRC with QApairs-I performs worse than the original BiDAF due to the accuracy of the upstream QApairs-I being only 0.69, which mislead in answering the current questions.

We directly used the bert-base-uncased model on Hugging Face to conduct the sentence coding without fine tuning in place of the sentence coding performed by BiLSTM. The results show that the large-scale pre-training model is effective for reading comprehension tasks. Meanwhile, the question comprehension introduced by QAPairs also enhances the performance of the original bert.

We also visualize the performance of the BiDAF and our PrA-MRC on each question category in Figure 7. The result shows our model outperforms the BiDAF in all categories of questions except *do* question. We consider the reason is that the dataset of upstream QApairs task contains CoQA. The answer to *do* question in CoQA is yes/no, rather than the span extraction in SQuAD, and thus damages the performance of the model. However, it also demonstrates that the knowledge learned in the upstream task is transferred to the MRC model. For simple question like *when*, *who*, *which*, both models perform well, and our model also improve a little. For the *why* question, the performance of our model has improved a lot which indicates our QApairs model learned useful information for descriptive questions.

5.2.3 Case visualization study

We analyze several question cases that how the BiDAF model and our PrA-MRC model capture the most attended words in context as the answer to current question. The visualization of attention matrices for dual-query and context tuples in dual-query and context interaction layer is shown in Figure 8. For the question '*Which team won Super Bowl 50?*', BiDAF captures the attended words for each word in question independently, with *team* matching *Denver*, *won* matching *champion*, *super* matching *super*, *bowl* matching *bowl*. However, PrA-MRC capture a global attention on *which team*, so all the words in question match *nfl* teams like *Denver*, *Panthers*. For another example, "*Where did the Normans and Byzantines sign the peace treaty?*", BiDAF still focus on words that are closed to the individual word in question, but PrA-MRC generates prior and global attention for *where* and captures most of place nouns in the article like *Gllavenica*, *Deabolis*, *Kanina* and *Petrela*. Therefore, our model fuses the prior question-type information into the MRC model through a pretrained specific model which enhances the focus of traditional MRC models on question related text.

6 Conclusion

In this article, we develop a simple dataset, namely QApairs, which is derived from the previous released SQuAD, QuAC and CoQA datasets. We present a new question and answer matching task on QApairs for learning the implicit information sought by different kinds of questions without given context. And we integrate the prior information learned by QApairs to a span-based machine reading comprehension model BiDAF to improve the performance. We use two bi-attention flows mechanism between context and dual query to obtain query-aware and prior query-aware context representations. The experimental evaluations in Stanford Question Answering Dataset (SQuAD) show that our model has an improvement over the baseline model and our model show a certain

interpretability. The visualizations and discussions show that our dual query learned a suitable representation and is capable for answering many types of question. We insist that question answering task would be greatly improved if there were more elaborate question and answer pairs to provide more reliable prior knowledge.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding author.

Author contributions

YZ contributed to the conception of the study and performed the experiment and wrote the manuscript. BS helped perform the analysis with constructive discussions. XC helped perform the experiment.

Funding

This research was funded by the Fundamental Research Funds for the Central Universities (Grant number 2020YJS012).

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

1. Chen D. *Neural reading comprehension and beyond*. Stanford University (2018).

2. Richardson M, Burges CJ, Renshaw E. 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 (2013). p. 193–203.

3. Hermann KM, Kocisky T, Grefenstette E, Espeholt L, Kay W, Suleyman M, et al. Teaching machines to read and comprehend. *Adv Neural Inf Process Syst* (2015) 28:1693–701.

4. Rajpurkar P, Zhang J, Lopyrev K, Liang P. Squad: 100,000+ questions for machine comprehension of text (2016). arXiv preprint arXiv:1606.05250.

5. Trischler A, Wang T, Yuan X, Harris J, Sordoni A, Bachman P, et al. Newsqa: A machine comprehension dataset (2016). arXiv preprint arXiv:1611.09830.

6. Reddy S, Chen D, Manning CD. Coqa: A conversational question answering challenge. *Trans Assoc Comput Linguistics* (2019) 7:249–66. doi:10.1162/tacl_a_00266

7. Peters ME, Neumann M, Iyyer M, Gardner M, Clark C, Lee K, et al. Deep contextualized word representations (2018). arXiv preprint arXiv:1802.05365.

8. Radford A, Narasimhan K, Salimans T, Sutskever I. *Improving language understanding by generative pre-training* (2018).

9. Devlin J, Chang MW, Lee K, Toutanova K (2018). Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805.

10. Zhang J, Zhu X, Chen Q, Dai L, Wei S, Jiang H. Exploring question understanding and adaptation in neural-network-based question answering (2017). arXiv preprint arXiv:1703.04617.

11. Tayyar MH, Lee M, Barnden J. *Integrating question classification and deep learning for improved answer selection*. Santa Fe, NM: Association for Computational Linguistics (2018).

12. Conneau A, Kiela D, Schwenk H, Barrault L, Bordes A. Supervised learning of universal sentence representations from natural language inference data (2017). arXiv preprint arXiv:1705.02364.

13. Bowman SR, Angeli G, Potts C, Manning CD. A large annotated corpus for learning natural language inference. In: Proceedings of EMNLP (2015).

14. Lin T, Wang Y, Liu X, Qiu X. A survey of transformers (2021). arXiv preprint arXiv:2106.04554.

15. Yang Z, Dai Z, Yang Y, Carbonell J, Salakhutdinov RR, Le QV. Xlnet: Generalized autoregressive pretraining for language understanding. *Adv Neural Inf Process Syst* (2019) 32.

16. Seo M, Kembhavi A, Farhadi A, Hajishirzi H. Bidirectional attention flow for machine comprehension (2016). arXiv preprint arXiv:1611.01603.

17. Huang HY, Zhu C, Shen Y, Chen W. Fusionnet: Fusing via fully-aware attention with application to machine comprehension. *arXiv preprint arXiv: 1711.07341* (2017) 33

18. Zhang Z, Wu Y, Zhou J, Duan S, Zhao H, Wang R. SG-Net: Syntax-guided machine reading comprehension. *Proc AAAI Conf Artif Intelligence* (2020) 34(05): 9636–43. doi:10.1609/aaai.v34i05.6511

19. Huang HY, Choi E, Yih WT. Flowqa: Grasping flow in history for conversational machine comprehension (2018). arXiv preprint arXiv:1810.06683.

20. Yeh YT, Chen YN. FlowDelta: Modeling flow information gain in reasoning for conversational machine comprehension (2019). arXiv preprint arXiv: 1908.05117.

21. Qu C, Yang L, Qiu M, Croft WB, Zhang Y, Iyyer M. BERT with history answer embedding for conversational question answering. In: Proceedings of the 42nd international ACM SIGIR conference on research and development in information retrieval (2019). p. 1133–6.

22. Qu C, Yang L, Qiu M, Zhang Y, Chen C, Croft WB, Iyyer M. Attentive history selection for conversational question answering. In: Proceedings of the 28th ACM International Conference on Information and Knowledge Management (2019). p. 1391–400.

23. Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, et al. A comprehensive survey on transfer learning. *Proc IEEE* (2020) 109(1):43–76. doi:10.1109/jproc.2020.3004555

24. Deng J, Dong W, Socher R, Li LJ, Li K, Fei-Fei L. Imagenet: A large-scale hierarchical image database. In: Proceedings of the 2009 IEEE conference on computer vision and pattern recognition. IEEE (2009). p. 248–55.

25. Taigman Y, Yang M, Ranzato MA, Wolf L. Deepface: Closing the gap to human-level performance in face verification (2014). Proceedings of the IEEE conference on computer vision and pattern recognition. 1701.

26. Antol S, Agrawal A, Lu J, Mitchell M, Batra D, Zitnick CL, Parikh D. Vqa: Visual question answering. In: Proceedings of the IEEE international conference on computer vision (2015). p. 2425–33.

27. Jiang Y, Wu S, Gong J, Cheng Y, Meng P, Lin W, et al. Improving machine reading comprehension with single-choice decision and transfer learning (2020). arXiv preprint arXiv:2011.03292.

28. Kundu S, Lin Q, Ng HT. Learning to identify follow-up questions in conversational question answering. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (2020). p. 959–68.

29. Choi E, He H, Iyyer M, Yatskar M, Yih WT, Choi Y, Zettlemoyer L. Quac: Question answering in context (2018). arXiv preprint arXiv:1808.07036.

30. Yoon K. *Convolutional neural networks for sentence classification*. Doha, Qatar: EMNLP (2014).

31. Pennington J, Socher R, Manning CD. Glove: Global vectors for word representation. In: Proceedings of the 2014 conference on empirical methods in natural language processing. EMNLP (2014). p. 1532–43.

32. Srivastava RK, Greff K, Schmidhuber J. Highway networks (2015). arXiv preprint arXiv:1505.00387.

33. Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. *J machine Learn Res* (2014) 15(1):1929–58.

34. Zeiler MD. Adadelta: An adaptive learning rate method. arXiv preprint arXiv: 1212.5701 (2012).

35. Yu Y, Zhang W, Hasan K, Yu M, Xiang B, Zhou B. End-to-End Answer Chunk Extraction and Ranking for Reading Comprehension. arXiv preprint arXiv: 1610.09996 (2016).

36. Yang Z, Dhingra B, Yuan Y, Hu J, Cohen WW, Salakhutdinov R. Words or Characters? Fine-grained Gating for Reading Comprehension. arXiv preprint arXiv: 1611.01724 (2017).

37. Wang S, Jiang J. Machine comprehension using match-lstm and answer pointer. arXiv preprint arXiv:1608.07905 (2016).

38. Xiong C, Zhong V, Socher R. Dynamic coattention networks for question answering. arXiv preprint arXiv:1611.01604 (2016).

39. Correia AH, Silva JL, Martins TDC, Cozman FG. A fully attention-based information retriever. In 2018 International Joint Conference on Neural Networks (IJCNN) (2018). 1–8. (IEEE)