QASA : advanced document retriever for open-domain question answering by learning to rank question-aware self-attentive document representations

Nguyen, Trang M.; Tran, Van-Lien; Can, Duy-Cat; Ha, Quang-Thuy; Vu, Ly T.; Chng, Eng-Siong

2019

Nguyen, T. M., Tran, V.-L., Can, D.-C., Ha, Q.-T., Vu, L. T., & Chng, E.-S. (2019). QASA : advanced document retriever for open-domain question answering by learning to rank question-aware self-attentive document representations. Proceedings of the 3rd International Conference on Machine Learning and Soft Computing, 221-225.
doi:10.1145/3310986.3310999

<https://hdl.handle.net/10356/144883>

<https://doi.org/10.1145/3310986.3310999>

© 2019 ACM. All rights reserved. This paper was published in 2019 Proceedings of the 3rd International Conference on Machine Learning and Soft Computing and is made available with permission of ACM.

Downloaded on 26 Apr 2023 23:37:20 SGT

QASA: Advanced Document Retriever for Open-Domain Question Answering by Learning to Rank

Question-Aware Self-Attentive Document Representations

Trang M. Nguyen^{*†}, Van-Lien Tran^{*}, Duy-Cat Can^{*†}, Quang-Thuy Ha^{*},
Ly T. Vu[‡], Eng-Siong Chng^{†‡}

^{*} Faculty of Information Technology, University of Engineering and Technology, VNUH, Vietnam

[†] Temasek Laboratories @ NTU, Nanyang Technological University, Singapore

[‡] School of Computer Science and Engineering, Nanyang Technological University, Singapore

{trangnm, lientv, catcd, thuyhq}@vnu.edu.vn, {tlvu, ASESChng}@ntu.edu.sg

ABSTRACT

For information consumers, being able to obtain a short and accurate answer for a query is one of the most desirable features. This motivation, along with the rise of deep learning, has led to a boom in open-domain Question Answering (QA) research. While the problem of machine comprehension has received multiple success with the help of large training corpora and the emergence of attention mechanism, the development of document retrieval in open-domain QA is lagged behind. In this work, we propose a novel encoding method for learning question-aware self-attentive document representations. By applying pair-wise ranking approach to these encodings, we build a Document Retriever, called QASA, which is then integrated with a machine reader to form a complete open-domain QA system. Our system is thoroughly evaluated using QUASAR-T dataset and shows surpassing results compared to other state-of-the-art methods.

CCS Concepts

• **Information systems** → **Retrieval models and ranking**; *Question answering* • **Computing methodologies** → Natural language processing

Keywords

Open-domain Question Answering; Document Retrieval; Learning to Rank; Self-attention mechanism.

1. INTRODUCTION

Ever since being promoted by the TREC-8 track [19], open-domain Question Answering (QA) has gained increasing attention over the years. In general, users expect QA systems to be able to produce concise and appropriate answers as opposed to search engines which return a list of documents [14]. As the result, open-domain QA can be further divided into two subtasks, i.e. document retrieval and machine comprehension. While there are numerous models proposed for dealing with machine comprehension task [3, 6, 17], document retrieval in open-domain QA has not received much investigation; even though, a poor retriever can cause a serious bottleneck for the entire system as the machine reader depends on its output. Having acknowledged the problem, Dhingra et al. proposed QUASAR dataset [7] which encourages open-domain QA research to go beyond understanding a given document and be able to retrieve relevant documents from a large corpus provided only the question. Following this progression and the works in [3, 20], we focus on building an advanced model for document retrieval.

Taking the cues from various reading comprehension models that adopt attention mechanism and produce excellent results [6, 21],

we employ a self-attentive network [4] over question-aware document representations for our retriever, namely QASA. Then, we construct a neural-based scoring function to rank the documents using pair-wise learning to rank [2]. In this paper, our contributions are as follow:

- We propose a method for learning question-aware self-attentive document encoding that, to the best of our knowledge, is the first to be applied in document retrieval.
- We utilize the reader from DrQA [3] and combine it with our retriever to form a pipeline system for open-domain QA.
- Our system is thoroughly evaluated on QUASAR-T dataset [7] and achieves exceeding performance compared to other state-of-the-art methods.

2. RELATED WORK

Unlike closed-domain QA, which is restricted to a certain domain and requires manually constructed knowledge bases, open-domain QA aims to answer questions about basically anything [14]. Hence, it relies on world knowledge in the form of large corpora, e.g. Wikipedia. Many datasets have been proposed, such as SQuAD [16], WikiReading [10], or recently, QUASAR dataset [7], that facilitate the development of open-domain QA systems.

Thanks to the advance of deep learning, especially, the emergence of attention mechanism [1], there has been momentous progress in machine reading comprehension task. In [3], the reader comprises of a paragraph encoding layer which is a multi-layer bi-directional long short-term memory (BiLSTM) applied on a selective feature set, a question encoding layer that learns a single vector representation of the question, and two classifiers trained independently for predicting the boundaries of the answer span.

Cui et al. [5] propose Attention-over-Attention (AoA) reader in which another attention layer is introduced on top of document-level attention over individual query words. Dhingra et al. [6] use a bi-directional Gated Recurrent Unit (Bi-GRU) for question/document encodings at the beginning of each layer in their multi-hop architecture. Then, they apply a Gated-Attention module for each token of the sequence. Another mechanism, called Gated Attention-based recurrent networks, is employed by Wang et al. [21] prior to a self-matching layer which also uses attention to extract important evidence from the documents.

Although they have been shown to be highly efficient, in open-domain QA setting, these reading comprehension models depend heavily on document retrieval to acquire relevant documents. For example, the reading accuracy (exact match) of GA model [6] on

QUASAR-T test set is 60% but when considering the retriever’s performance, the overall accuracy drops to 26.4% [7]. Therefore, the focus is now shifting to improving document retrieval process [3, 20]. While the Document Retriever in [3] is relatively simple, R³ ranker [20] is jointly trained with the reader using reinforcement learning to form an end-to-end open-domain QA system. R³ has state-of-the-art results in both document retrieval and machine comprehension task.

Inheriting the idea of using attention mechanism to learn the representation of documents [4, 21] and pair-wise learning to rank approach [2], we develop an advanced document retriever that empowered by ranking question-aware self-attentive (QASA) document representations. We then integrate it with the Document Reader from [3] to have a complete open-domain QA system that can be thoroughly evaluated and compared with other methods.

3. PROBLEM STATEMENT

Open-domain QA systems usually comprise of two modules: a Document Retriever and a Document Reader [3]. Given a question q , the Document Retriever acquires top- k documents from a search space by ranking them based on their relevance to q . Let D represent all documents in the search space, the set of top- k highest scored documents is:

$$D^* = \operatorname{argmax}_{x \in |D|^k} \left(\sum_{d \in x} f(d, q) \right) \quad (1)$$

where $f(\cdot)$ is the scoring function. The Document Reader takes q and D^* as input and produces an answer a which is a text span in some $d_j \in D^*$ that gives the maximum likelihood of satisfying the question q .

4. MATERIAL AND METHODS

Following the conventional pipeline, our system consists of a Document Retriever and a Document Reader. The Document Retriever is an end-to-end deep learning model that can be divided further into four components: (1) an Embedding Layer for mapping each word in the questions and documents into a vector space, (2) a Question Encoding Layer and (3) a Document Encoding Layer that produce the final representations of the questions and documents, respectively, and (4) a neural-based Scoring Function for learning an effective similarity measurement between two fixed-size vectors. To exploit the power of our Document Retriever in an open-domain QA setting, we utilize the Document Reader from DrQA [3] for extracting the answer from retrieved documents.

4.1 Document Retriever

In several previous works [17, 18, 21] on reading comprehension task, the information from the question and document are fused together to form question-aware document representations. Inspired by this idea and the advance of attention mechanisms, the final document encoding of our model is produced by applying a simple self-attention mechanism that conditioned not only on the document itself but also on the question encoding. We hypothesize that this question-aware self-attentive document encoding layer will learn better representations than the one which does not take the question information into account. Our Document Retriever is depicted in Figure 1.

4.1.1 Embedding Layer

An embedding layer (EL) is commonly used as the first layer in a deep learning model in order to solve various natural language processing (NLP) problems [22]. It assigns a distributional vector

Figure 1. The architecture of the Document Retriever.

to each token in the input sequence which can be further processed by subsequent layers. Our model uses token-level and character-level embeddings to capture both semantic and morphological information of words. All parameters in this layer are shared between questions and documents to maximize the representation power.

Token Embedding: After the pre-processing step, each token is mapped to its embedding by the mean of a look-up table. We use the pre-trained English word vectors from fastText [9], in which they employ continuous bag-of-words (CBOW) [15] with position-weights.

Character Embedding: The use of character embedding has been applied by many other works [6, 17, 21] for its capability of handling out-of-vocabulary (OOV) problem. In this paper, let V_C be the character set, the character embedding matrix $\mathbf{C} \in \mathbb{R}^{|V_C| \times n}$ are first created randomly by Glorot initialization [8] and then fine-tuned as trainable parameters of the model. For each token t , using a look-up table, we can obtain a sequence of character embeddings $\mathbf{T} = \{c_1, c_2, \dots, c_{|T|}\}$, $c_i \in \mathbf{C}$. A single layer of BiLSTM [11] is then applied on $\mathbf{T}$ to produce character-level embedding e^c :

$$e^c = \vec{h}_{|T|} \oplus \bar{h}_{|T|} \quad (2)$$

where $\oplus$ is the concatenating function; $\vec{h}_{|T|}$ and $\bar{h}_{|T|}$ are the last hidden states of the forward and backward direction, respectively.

Given a sequence of tokens $P = \{t_i\}_{i=1}^{|P|}$, which is either a question or a document, the output of EL is a sequence of embeddings $\mathbf{E} = \{e_i\}_{i=1}^{|P|}$, in which:

$$e_i = e_i^t \oplus e_i^c \quad (3)$$

where e_i^t is the pre-trained token embedding, and e_i^c is the character embedding of t_i .

4.1.2 Question Encoding Layer

This layer aims to learn the contextual representation of the question by applying one BiLSTM layer followed by two fully-connected layers to $\mathbf{E}^q = \{\mathbf{e}_i^q\}_{i=1}^{|Q|}$, the output of EL, with $|Q|$ is the question's length:

$$\bar{\mathbf{H}}^q = \overrightarrow{\text{LSTM}}(\mathbf{E}^q) = \{\bar{\mathbf{h}}_i^q\}_{i=1}^{|Q|} \quad (4)$$

$$\tilde{\mathbf{H}}^q = \overleftarrow{\text{LSTM}}(\mathbf{E}^q) = \{\tilde{\mathbf{h}}_i^q\}_{i=1}^{|Q|} \quad (5)$$

$$\mathbf{h}^q = \bar{\mathbf{h}}_{|Q|}^q \oplus \tilde{\mathbf{h}}_{|Q|}^q \quad (6)$$

$$\mathbf{a}^q = \text{ReLU}(\mathbf{W}^{(1)}\mathbf{h}^q + \mathbf{b}^{(1)}) \quad (7)$$

$$\mathbf{q}_e = \mathbf{W}^{(2)}\mathbf{a}^q + \mathbf{b}^{(2)} \quad (8)$$

where $\mathbf{h}^q$ is the concatenation of the last hidden states; $\mathbf{W}^{(1)}$, $\mathbf{W}^{(2)}$, $\mathbf{b}^{(1)}$, and $\mathbf{b}^{(2)}$ are trainable weights and biases of the model. After this step, we obtain the encoding, namely $\mathbf{q}_e$, of the input question.

4.1.3 Document Encoding Layer

For humans, in order to decide whether a document is relevant to a particular question, the meaning of that question should always be kept in mind while we read. Following this intuition, this layer utilizes the question encoding $\mathbf{q}_e$, which embodied the general meaning of the question, to form question-aware hidden states, $\bar{\mathbf{H}}$, by concatenating $\mathbf{q}_e$ with the hidden states outputted by a BiLSTM as follow:

$$\begin{aligned} \mathbf{H}^d &= \overrightarrow{\text{BiLSTM}}(\mathbf{E}^d) \\ &= \{\bar{\mathbf{h}}_i^d \oplus \tilde{\mathbf{h}}_{|D|-i+1}^d\}_{i=1}^{|D|} \end{aligned} \quad (9)$$

$$= \{\mathbf{h}_i^d\}_{i=1}^{|D|}$$

$$\bar{\mathbf{H}} = \{\mathbf{h}_i^d \oplus \mathbf{q}_e\}_{i=1}^{|D|} = \{\bar{\mathbf{h}}_i\}_{i=1}^{|D|} \quad (10)$$

where $|D|$ is the number of tokens in the document; $\mathbf{E}^d = \{\mathbf{e}_i^d\}_{i=1}^{|D|}$ are the word embeddings from EL. To take advantage of the contextual representing power, we use the same BiLSTM for both question and document encoding layer.

We then apply a simple self-attentive network [4] to $\mathbf{H}^d$ where the attention weights are calculated based on $\bar{\mathbf{H}}$:

$$\mathbf{a}_i^d = \text{ReLU}(\mathbf{W}\bar{\mathbf{h}}_i + \mathbf{b}) \quad (11)$$

$$\mathbf{u} = \{\mathbf{v} \cdot \mathbf{a}_i^d + b\}_{i=1}^{|D|} \quad (12)$$

$$\boldsymbol{\alpha} = \text{softmax}(\mathbf{u}) \quad (13)$$

where $\mathbf{W} \in \mathbb{R}^{m \times (2z+l)}$, $\mathbf{v}, \mathbf{b} \in \mathbb{R}^m$, and $b \in \mathbb{R}$ are parameters, with z and m are the numbers of the hidden units in the BiLSTM and fully-connected layer respectively, l is the question encoding size. Final representation of the document $\mathbf{d}_e$ is the linear transformation of $\mathbf{c}_e$, which is the weighted sum of the hidden states $\mathbf{H}^d$:

$$\mathbf{c} = \sum_{i=1}^{|D|} \alpha_i \mathbf{h}_i^d \quad (14)$$

$$\mathbf{d}_e = \mathbf{W}\mathbf{c} + \mathbf{b} \quad (15)$$

where $\mathbf{W}, \mathbf{b}$ are learnable parameters, $\alpha_i \in \boldsymbol{\alpha}$ and $\mathbf{h}_i^d \in \mathbf{H}^d$. By integrating the question encoding with the document's hidden states as in Eq. 10, the document encoding is conditioned on the question, such that it allows the model to produce distinctive representations of the same document depends on the question given.

4.1.4 Scoring Function

Our scoring function is a neural network comprises of two feed-forward layers, which is similar to the idea of applying matching methods to extract relations between two vectors in [4]. We also find that [12] uses this approach for their Ranker except its input are pre-trained vectors. Since our Document Retriever can be trained in an end-to-end fashion, the error is backpropagated through the scoring function and the encoding layers, which enables the model to learn better similarity measurement as well as question/document representations simultaneously.

Much the same as [4], our feature vector for the scoring function is also a concatenation of the question encoding $\mathbf{q}_e$, document encoding $\mathbf{d}_e$, their element-wise product, and their absolute element-wise difference. Given the two encodings, their similarity score can be calculated as follow:

$$\mathbf{x} = \begin{bmatrix} \mathbf{q}_e \\ \mathbf{d}_e \\ \mathbf{q}_e \odot \mathbf{d}_e \\ |\mathbf{q}_e - \mathbf{d}_e| \end{bmatrix} \quad (16)$$

$$\mathbf{a} = \text{ReLU}(\mathbf{W}\mathbf{x} + \mathbf{b}) \quad (17)$$

$$s = \mathbf{w} \cdot \mathbf{a} + b \quad (18)$$

where $\odot$ is the Hadamard product; $\mathbf{W} \in \mathbb{R}^{r \times 4l}$, $\mathbf{w}, \mathbf{b} \in \mathbb{R}^r$, and $b \in \mathbb{R}$ are trainable parameters of the network (l is the encoding size, r is the number of hidden units); scalar s is the similarity score between $\mathbf{q}_e$ and $\mathbf{d}_e$.

4.1.5 Training Process

Let S be the function modeled by the Scoring Function, the Document Retriever model can be trained by minimizing the margin ranking loss [2]:

$$L = \max(0, 1 - S(\mathbf{q}_e, \mathbf{d}_e^+) + S(\mathbf{q}_e, \mathbf{d}_e^-)) \quad (19)$$

where $\mathbf{d}_e^+$ and $\mathbf{d}_e^-$ are the encodings of the positive and negative document, respectively. For the fact that we share the same problem as [20] where the ground-truth labels for the ranking task are not available, we employ their idea of *pseudo labels* and consider a document as positive if it contains the exact match of the answer.

4.2 Document Reader

As our paper focuses on improving the document retrieval process, we reuse the Document Reader from DrQA [3] to demonstrate how a better retriever can result in higher performance of the overall system. The training data for the reader is generated by selecting 50 highest scored documents ranked by the retriever, in which there are no more than 30 positive documents. We find that this combination of positive and negative documents in the training data boosts the reader's performance greatly since it resembles the testing phase when the outputted documents of the retriever are not entirely positive.

Table 1. QUASAR-T Statistics. “Valid” indicates the number of instances in which the ground-truth answer (after pre-processed) is presented in the pseudo-documents.

	Train	Validation	Test
Total	37,012	3,000	3,000
Valid	28,838	2,297	2,321

5. EXPERIMENTAL EVALUATION

5.1 Dataset

We train and test our models on the QUASAR-T dataset proposed by [7] using the official splits provided. This standard dataset consists of 43,012 factoid questions obtained from numerous sources. Each of the questions is associated with 100 pseudo-documents retrieved from ClueWeb09. The answers for those questions are free-form text spans, however, they are not guaranteed to appear in the documents which are challenging for both ranking and reading model. The statistics of QUASAR-T dataset are described in Table 1. According to this, the upper bound for evaluating the performance of the retriever and the reader is the ratio between the number of valid instances and the total number of instances. Particularly, for the test set, this upper bound is 77.37%.

5.2 Baseline Models

Our model is compared with six other proposed models that have results for the QUASAR-T dataset: GA [6], a reader that integrates a multi-hop architecture with attention mechanism for text comprehension; BiDAF [7], which uses bi-directional attention flow mechanism; InferSent Ranker and Relation-Networks (RN) Ranker [12], which are simple neural ranking functions, combined with DrQA [3]; R³ [20], a novel Ranker-Reader system that is trained using reinforcement learning, and its simpler version, SR² [20], trained by combining two different objective functions from the ranker and reader.

To demonstrate the effectiveness of learning QASA document representations, we take out the self-attention mechanism from the full model. That is, the Document Encoding Layer for this *kickout model* is constructed using the same architecture as the Question Encoding Layer.

5.3 Model Settings

We implement our models using TensorFlow. All parameters as well as the character embedding look-up table are randomly constructed using the Glorot normal initializer [8] and learned during the training phase. We use Adam optimization [13] with mini-batch gradient descent; the initial learning rate is set to 0.001 and the batch size is 32. The number of hidden units in the contextual BiLSTM and the character BiLSTM are 150 and 50 respectively. In the self-attentive network, the fully-connected layer has the size of 200 units, and for the scoring function, this number is 50 units. The encoding vectors q_e and d_e both have 128 dimensions.

During the training phase, we take into account all positive documents. Top-20 highest scored negative ones are selected at the beginning of each epoch w.r.t the current parameters of the model. This is to ensure that the model is trained against the most challenging instances where the positive and negative document are difficult to marginalize.

Table 2. Evaluation of retriever models on the QUASAR-T test set using *top-k* recall [20]. RN ranker[†] is the reimplemented version of RN ranker [12].

	Top-1	Top-3	Top-5
InferSent ranker [12]	36.10	52.80	56.70
RN ranker [12]	51.40	68.20	70.30
RN ranker [†]	27.23	41.90	50.27
SR ² ranker [20]	28.80	46.40	54.90
R ³ ranker [20]	40.30	51.30	54.50
QASA	32.67	47.93	54.90
Kickout model	32.43	46.57	53.20

Table 3. The overall performance of various open-domain QA systems evaluated on the QUASAR-T test set.

	F1	EM
BiDAF [17]	28.50	25.90
GA [6]	26.40	26.40
InferSent + DrQA [12]	37.60	31.20
RN + DrQA [12]	30.70	26.00
SR ² [20]	38.80	31.90
R ³ [20]	40.90	34.20
QASA + DrQA	45.07	40.50
Kickout model + DrQA	43.39	38.37

5.4 Results and Discussion

5.4.1 Evaluation Metrics

For the Document Retriever module, we employ the *top-k recall* metric [20], in which the top- k documents for a particular question are considered correctly retrieved if they include at least one positive document.

The overall system is evaluated using two widely used metrics which are F1 and Exact Match (EM) [16]. Specifically, F1 measures the overlap between two bags of tokens that correspond to the ground-truth and predicted answer while EM strictly compares the two strings to check whether they are equal or not.

5.4.2 Document Retriever Results

The results for our Document Retriever are presented in Table 2 as it is compared with four other models. We find that the original results for RN ranker from [12] are unexpectedly high regarding being a relatively simple model. Since their source code is unavailable, we make an attempt to reimplement the model to be as close as possible to its description in the paper; this version is denoted as RN ranker[†]. The fact that RN ranker[†]’s results are significantly lower than being reported in [12] casts doubt upon their evaluation process. For this reason, the original performance of InferSent and RN ranker is only addressed in Table 2 for referencing purpose without being considered reliable.

As mentioned, R³ [20] jointly trains the document retrieval and answer extraction module simultaneously using reinforcement learning. By the mean of the rewarding scheme, their ranker can gain some insight into the reader’s performance while being trained. This helps R³ mitigate the cascading error problem that most pipeline systems, including ours, suffer from and boosts its

recall remarkably. As the result, their ranker has much higher recall in top-1 and top-3 than QASA although being slightly lower in top-5. Another model from [20] is SR^2 which is a simpler variant of R^3 . Because SR^2 is not benefited by joint learning, its ranker is more comparable to our model. To this end, QASA shows more favorable results where it achieves 3.87% and 1.53% higher than SR^2 ranker in top-1 and top-3 respectively.

When comparing with our *kickout model*, which only uses a feed-forward neural network for document encoding, QASA also produces surpassing results among all top- k recall. Concretely, by using the QASA document representation, the model gives an improvement of 1.7% in top-5 which proves our hypothesis.

5.4.3 Overall Results

The overall results for several open-domain QA systems are summarized in Table 3. As can be seen from the table, QASA consistently offers better results than the *kickout model* when integrated with DrQA [3], which proves once again the effectiveness of question-aware self-attentive mechanism. Moreover, our system excels R^3 by a large margin (4.17% in F1 and 6.3% in EM) in spite of being designed in a pipeline manner.

It is worth noting that the QUASAR-T dataset does not provide ground-truth for document retrieval, therefore, this module is evaluated using *pseudo labels* described in 4.1.5. A limitation of *pseudo labels* is that the positive documents are not guaranteed to be relevant to the question. For example, given the question “What is the smallest state in the US?”, one of its positive documents is “1790, Rhode Island ratifies the United States Constitution and becomes the 13th US state” (it contains the answer, “Rhode Island”). However, this positive document does not help the reader since it is completely irrelevant. For the reader to extract the answer, not only the retrieved document must enclose the exact string but also it must convey information related to the query.

For that reason, even though our Document Retriever has lower recall than R^3 ranker, its outputted documents are semantically similar to the question, thus, they are more useful to the Reader which results in a much higher performance of the overall system.

6. Conclusions

We introduced a novel Document Retriever, namely QASA, which can rank documents by encoding them using question-aware self-attentive mechanism. We then constructed a pipeline system for open-domain QA by combining our retriever with the reader from DrQA. Having analyzed the results of QASA compared to the *kickout model*, we demonstrate the effectiveness of using question-aware self-attentive representations for document retrieval in open-domain QA. We also show that the retriever has a substantial contribution to the overall system and by improving the retriever, we can extend the upper bound of machine reading module markedly.

7. REFERENCES

- [1] Bahdanau, D.; Cho, K.; Bengio, Y. Neural machine translation by jointly learning to align and translate. *Proceedings of the ICLR*, 2015.
- [2] Bai, B. et al. Learning to rank with (a lot of) word features. *Information retrieval* 2010, 13 (3), 291-314.
- [3] Chen, D.; Fisch, A.; Weston, J.; Bordes, A. Reading Wikipedia to Answer Open-Domain Questions. *Proceedings of the ACL*, 2017; pp 1870-1879.
- [4] Conneau, A.; Kiela, D.; Schwenk, H.; Barrault, L.; Bordes, A. Supervised Learning of Universal Sentence Representations from Natural Language Inference Data. *Proceedings of the EMNLP*, 2017; pp 670-680.
- [5] Cui, Y.; Chen, Z.; Wei, S.; Wang, S.; Liu, T.; Hu, G. Attention-over-Attention Neural Networks for Reading Comprehension. *Proceedings of the ACL*, 2017; pp 593-602.
- [6] Dhingra, B.; Liu, H.; Yang, Z.; Cohen, W.; Salakhutdinov, R. Gated-Attention Readers for Text Comprehension. *Proceedings of the ACL*, 2017; pp 1832-1846.
- [7] Dhingra, B.; Mazaitis, K.; Cohen, W. W. Quasar: Datasets for Question Answering by Search and Reading. *arXiv preprint arXiv:1707.03904* 2017.
- [8] Glorot, X.; Bengio, Y. Understanding the difficulty of training deep feedforward neural networks. *Proceedings of the AISTATS*, 2010; pp 249-256.
- [9] Grave, E. et al. Learning Word Vectors for 157 Languages. *Proceedings of the LREC*, 2018.
- [10] Hewlett, D. et al. WikiReading: A Novel Large-scale Language Understanding Task over Wikipedia. *Proceedings of the ACL*, 2016; pp 1535-1545.
- [11] Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural computation* 1997, 9 (8), 1735-1780.
- [12] Htut, P. M.; Bowman, S.; Cho, K. Training a Ranking Function for Open-Domain Question Answering. *Proceedings of the NAACL: Student Research Workshop*, 2018; pp 120-127.
- [13] Kingma, D. P.; Ba, J. Adam: A method for stochastic optimization. *Proceedings of CoRR*, 2014.
- [14] Kolomiyets, O.; Moens, M.-F. A survey on question answering technology from an information retrieval perspective. *Information Sciences* 2011, 181 (24).
- [15] Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; Dean, J. Distributed representations of words and phrases and their compositionality. *Advances in neural information processing systems*, 2013; pp 3111-3119.
- [16] Rajpurkar, P.; Zhang, J.; Lopyrev, K.; Liang, P. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *Proceedings of the EMNLP*, 2016; pp 2383-2392.
- [17] Seo, M.; Kembhavi, A.; Farhadi, A.; Hajishirzi, H. Bidirectional Attention Flow for Machine Comprehension. *Proceedings of the ICLR*, 2017.
- [18] Shen, Y.; Huang, P.-S.; Gao, J.; Chen, W. Reasonet: Learning to stop reading in machine comprehension. *Proceedings of the KDD*, 2017; pp 1047-1055.
- [19] Voorhees, E. M. et al. The TREC-8 Question Answering Track Report. *Trec*, 1999; pp 77-82.
- [20] Wang, S. et al. R3: Reinforced ranker-reader for open-domain question answering. *Conference on Artificial Intelligence*, 2018.
- [21] Wang, W.; Yang, N.; Wei, F.; Chang, B.; Zhou, M. Gated self-matching networks for reading comprehension and question answering. *Proceedings of the ACL*, 2017.
- [22] Young, T.; Hazarika, D.; Poria, S.; Cambria, E. Recent trends in deep learning based natural language processing. *IEEE Computational intelligence magazine* 2018, 13 (3).