AQuPR: Attention based Query Passage Retrieval

Parth Pathak, Mithun Das Gupta, Niranjan Nayak, Harsh Kohli
Microsoft India Development Center
Hyderabad, India
papatha,migupta,niranjan,hakohl@microsoft.com

ABSTRACT

Search queries issued over the Web increasingly look like questions, especially as the domain becomes more specific. Finding good response to such queries amounts to finding relevant passages from Web documents. Traditional information retrieval based Web search still matches the query to the words in the entire document. With the advent of machine reading comprehension techniques, Web search is moving more towards identifying the best sentence / group of sentences in the document. We present AQuPR an **A**ttention based **Qu**ery **P**assage **R**etrieval system to find human acceptable answer containing passages to technology queries issued over the Web. We train character level embeddings for the query and passage pairs, train a deep recurrent network with a novel simplified attention mechanism and incorporate additional signals present in Web documents to improve the performance of such a system. We collect a database of human issued queries along with their answer passages and learn an end to end system to enable automated query resolution. We present results for answering human issued search queries which show considerable promise against basic versions of current generation question answering systems.

CCS CONCEPTS

• **Information systems** → *Clustering and classification*; *Page and site ranking*;

KEYWORDS

Question Answering; Passage retrieval; Attention mechanism; Char-CNN; Highway Networks.

ACM Reference Format:
Parth Pathak, Mithun Das Gupta, Niranjan Nayak, Harsh Kohli. 2018. AQuPR: Attention based Query Passage Retrieval. In *2018 ACM Conference on Information and Knowledge Management (CIKM'18), October 22–26, 2018, Torino, Italy.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/No:sp1853

Web searching is aggressively moving towards question answering (QA) systems. Building intelligent agents with the ability for reading comprehension (RC) or open-domain question answering (QA) over real world data is a major goal of artificial intelligence. QA can be thought of a more specific case of sentence answering (SA) [11] which tries to model two semantically related sentences together. Consequently, RC and QA from Web data have become very promising research fields, especially with the introduction of MS-MARCO [2] and SQuAD [4] datasets.

CIKM '18, October 22–26, 2018, Torino, Italy

ACM ISBN 978-1-4503-6014-2/18/10…$15.00
https://doi.org/10.1145/No:sp1853

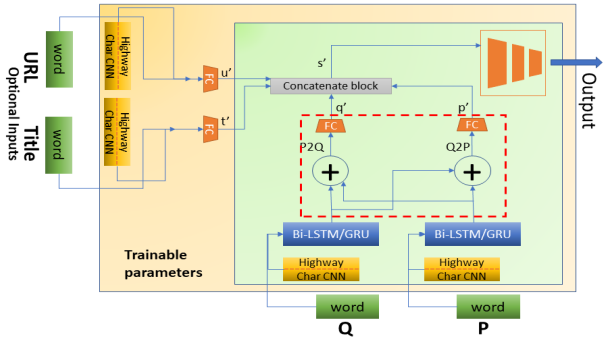

Figure 1: Deep architecture for AQuPR.

One of the key factors for the advancement of sequence to sequence models such as SA, RC and QA, has been the use of neural attention mechanism. Attention mechanism enables the system to focus on a targeted area within a context paragraph that is most relevant to answer the question [9].

QA tries to predict the exact answer span from a given string. This method, though extremely powerful is sometimes too limiting for generic web queries and does not scale well with the number of passages. Showing the entire URL is also limiting, since the user has to read the entire document to arrive at the answer. We propose to solve a problem at the middle of these two ends, wherein the best 'passage' is identified from the document, which is highly likely to contain the answer that the user is looking for. We define a contiguous chunk of k words from a document as a passage. The passages can have a small overlap of m words. The output of our system can then be fed to a RC system to find the correct span. This idea is the opposite of the RC result aggregation work recently reported in [7].

In this paper, we translate the passage retrieval problem to web query based passage retrieval problem based on the idea of "reading and selecting". We take up a specific segment of all web queries, the technology intent segment, and design a bi-directional attention based network to answer web queries.

1 AQuPR FOR TECH QUERIES

The attention based query passage retrieval (AQuPR) architecture proposed in this work is shown in Fig. 1. Input to the system is query (**Q**), passage (**P**), along with optional inputs which are title **T** of the Web document and its **URL**. The output is a scoring for the passage based on the notion of whether the passage is a possible response for the given query. For tokenizing the URL we remove

all the special characters from it and then treat all the remaining entities as valid strings which are then passed through embedding techniques. The output of the system is a scoring for the passages and hence we do not put a recurrent neural net based decoder but replace it with a compressing cascade of fully connected layers.

1.1 Input Query and Passage Embedding

Assume that our vocabulary has $|V|$ words. Word embedding technique transforms a sequence of w words (e.g. a sentence) denoted as $\mathbf{x}$ into a fixed dimensional representation given by

$$\mathbf{x} \in \mathbb{Z}^{w \times |V|} \underbrace{\rightarrow}_{\text{embedding}} \mathbb{R}^{w \times e} \tag{1}$$

where e is the embedding dimension. In this work, for each word in the input query, passage, title and URL, we obtain its corresponding GloVe [3] embedding with $e = 300$ dimensions.

Character based word embedding has been shown to outperform word based model for language modeling. This is especially true for domain specific words which constitute the heavy tail of the token frequencies collected from Web documents [1]. We collected over 1M unique pair of query and documents, tokenized them using natural language tool kit (http://www.nltk.org), We project these input tokens through the charCNN layer followed by the highway network to learn the character level embedding for the words present in the input data. For an input word w_i of $c \leq c_{\max}$ characters the encoding works as follows:

$$\mathbf{w_i} \underbrace{\rightarrow}_{\text{CharCNN encoding}} \mathbb{R}^{c_{\max} \times 20} \tag{2}$$

where $c_{\max}$ is the maximum number of characters in the word and we learn 20 dimensional embedding for each character. For words with lesser characters in them, appropriate zero padding is applied. We pass this encoded matrix through five 1-D convolution layers of size $\{C_1, C_2, \ldots, C_5\} \times 20$ and finally through a highway network [6].

Similar to the memory cells in LSTM networks, highway layers allow for training of deep networks by adaptively carrying some dimensions of the input directly to the output. The output of the highway network is fixed to 200 dimensions. This output is then appended to the embedding obtained from GloVe to generate an embedding of 500 dimensions. Empirically, retraining the GloVe vectors with our data leads to over-fitting and the resultant models do not generalize well. Keeping a combination of pre-trained Glove vectors with trained charCNN vectors results in the best test performance.

For the optional inputs to our model, namely URL and title, we add a fully connected network to bring the dimension of the embedding down to 50. The core inputs, the query and the passage, are passed through bi-directional LSTMs for semantic encoding. We treat these inputs essentially as filters and do not deploy high cost LSTMs to model these.

We implement a dynamically unfolding LSTM for encoding the sequence of words into meaningful representations. For an input entity, the biLSTM model encodes each input word w_k into a $2d$ dimensional vector denoted as

$$\mathbf{h_{k_F}} = \mathtt{LSTM}_F(w_k, h_{k-1}|_{w_{k-1}}) \in \mathbb{R}^d \tag{3}$$

$$\mathbf{h_{k_B}} = \mathtt{LSTM}_B(w_k, h_{k+1}|_{w_{k+1}}) \in \mathbb{R}^d \tag{4}$$

where $d = 50$ in all our experiments, $h_k|_{w_j}$ represents the k^{th} hidden representation obtained after processing the text entity upto the j^{th} word in the forward or backward direction as determined from the context. Every word is finally represented as a concatenation of the forward and backward encoding denoted as $w_k \rightarrow \mathbf{w}_k = [\mathbf{h_{k_F}}; \mathbf{h_{k_B}}] \in \mathbb{R}^{2d}$. Based on this encoding query $\mathbf{q}$ is encoded as a matrix of size $w_q \times 100$, passage $\mathbf{p}$ is encoded as a matrix of size $w_p \times 100$ and similar encodings are obtained for title and the URL.

1.2 Attention Module

The output of the bi-LSTM stage are individual embeddings for the query as well as the passage words. We implement a simple attention module to weigh the query words with respect to the passage (P2Q) and passage words with respect to the query (Q2P). Bi-directional attention flow for machine comprehension was recently proposed by [5]. Along the same lines, in the R-net work [8], the authors propose a similar model where they learn a non-linearity for the attention of the k_{th} word based on the attention of the $k - 1_{th}$ word in the passage.

Our hypothesis in this work is to adopt a simpler attention module similar to the model presented by Yin et al. [11]. We generate a unified attention matrix $\mathbf{A}$ denoted as

$$\mathbf{A}_{w_q \times w_p} = \mathbf{q}_{w_q \times 100}.\text{Diag}(\mathbf{a}_{\mathbf{w}100}).\mathbf{p}^T_{100 \times w_p} \tag{5}$$

where $\mathbf{a}_{\mathbf{w}100}$ is a trainable attention vector equal in size to the embedding dimension of 100, and Diag(.) creates a diagonal matrix. The attention **Q2P** from query $\mathbf{q}$ to passage $\mathbf{p}$ is obtained as

$$\mathbf{Q2P} = \mathbf{1}^T_{w_q} \mathbf{A}_{w_q \times w_p} \tag{6}$$

$$\mathbf{p}' = \text{FC}_{w_P, 50}(\mathbf{Q2P}) \in \mathbb{R}^{50} \tag{7}$$

where $\mathbf{1}_k$ is a column vector of all ones of size k and $\text{FC}_{m,n}$ represent a feed forward fully connected layer of input size m and output size n. $\mathbf{p}'$ denotes the query waited passage representation for future processing. Similar to **Q2P**, the attention from passage $\mathbf{p}$ to query $\mathbf{q}$ is obtained by

$$\mathbf{P2Q} = \mathbf{A}_{w_q \times w_p} \mathbf{1}_{w_p} \tag{8}$$

$$\mathbf{q}' = \text{FC}_{w_q, 50}(\mathbf{P2Q}) \in \mathbb{R}^{50} \tag{9}$$

As in the case of the passage, $\mathbf{q}'$ denotes the passage weighted query representation for future processing. We would like to stress here that, unlike most of the recent papers in this field we maintain equal importance for both the query representation and the passage representation, as evident from the equal dimensional (50 in our experiments) representation for both. Note that Yin et al. [11] do not have a trainable weight as the inner product while computing the attention matrix (Eq. 5). They learn two different weight matrices for the row/column sums as shown in Eq. 6 and Eq. 8.

The output from the attention layers are combined with the encoding of the title of the document as well as the URL to generate the input for the final classification module which is a cascade of fully connected feed forward networks.

2 EXPERIMENTS AND RESULTS

The simplified attention mechanism mentioned in this work is very similar to the attention block proposed by Yin et al. [11]. We introduce learnable weights for the feature dimension of the embeddings, which has been proved to be important to capture inter feature interactions [5]. Note that Yin et al. [11] have a CNN based technique which is entirely different from our model. Keeping this in mind we compare with their method only for the very small WikiQA dataset [10] as shown in Table. 1. The WikiQA dataset consists of 20,360 question candidate pairs for training, 1,130 pairs in dev and 2,352 pairs for test. Our method is not restricted to limiting the answer passage and hence we keep the entire passage, as compared to [10, 11] who truncate the answer to 40 tokens.

Model	MAP	AUC
BCNN	0.6378	0.7280
ABCNN1	0.6414	0.7312
AQuPR	0.5843	0.7536

Table 1: Comparison results on WikiQA dataset for the proposed attention model AQuPR against the similar attention models BCNN ans ABCNN1 in [11].

2.1 Web Query Data

We start with an initial set of around 2.3M queries and their top 10 web pages as returned by Bing search engine. We collect the URL, title and the document body for each of these retrieved passages. We perform topic modeling on the queries and documents to identify the most influential topics in these results. Once the topics are identified we perform a related keyword search for each topic within the dataset. We define related keywords as those words which appear with high probability with the topic words within a document. This gives a wider representation of each topic. We count the frequencies of each topic and related keywords and smooth the frequency distribution. All the queries are projected onto the topic distributions and then clustered by k-Means clustering algorithm. For our experiments we maintain close to 40 unique topic distributions. Finally, we perform weighted sampling from the query clusters to identify the final set of around 50K queries which we use for training our model.

Once this smaller representative set of queries and their top 10 links are collected, we divide the linked documents into passages with parameters $[k, m] = [250, 100]$ with conditional overlap. For passages with ending close to sentence ending/beginning we adjust the overlap within 100 words to accommodate the sentence end/begin. For any query, one unique passage from one unique document is selected as the answer containing passage based on aggregation of votes of independent human judges. We asked 3 independent human judges to score each query in our set.

Once the positive passage for a query is identified, we perform a character W-shingling[1] based cosine similarity search with the positive passage against all the other passages obtained from the documents identified by the search engine for the same query. We compute the cosine similarity with the positive passage and rank all the negative passages in decreasing order of similarity. We select upto 3 random negative passages from the lower third quantile of the scores. We generated a dataset of $\approx$ 200K query passage pairs.

[1]https://en.wikipedia.org/wiki/W-shingling

Model	**Embedding**	**AUC Score**
BDT	-	0.62
SVM	-	0.65
BiDAF	GloVe + Char	0.9112
R-Net	GloVe + Char	0.9144
AQuPR	GloVe + Char	0.9453
AQuPR + Op	**GloVe + Char**	**0.9661**

Table 2: Experiment results. BDT stands for Gradient Boosted Decision Trees, SVM is support vector machine, AQuPR is the proposed method. 'Op' denotes optional inputs.

We compare our work with two main works which are similar in design to ours but solve different problems. The BiDAF model [5] and the R-Net model [8]. Both these models are similar to the proposed model AQuPR as presented in the preceding sections. We further compare our approach with two baselines - a) Support Vector Machine (SVM) [23] based classifier and b) Gradient Boosted Decision Trees (BDT) [24] based classifier. We use a bag-of-words model over the query words as features and the labeled training set for training. We report the best results (AUC scores) obtained from these models after parameter tuning in Table. 2. For all the results the hyper-parameter set $\{w_q, w_p, w_t, w_u, c_{\max}\}$ was empirically determined and then held fixed at $\{15, 250, 15, 5, 20\}$ respectively. The convolution layers for the CharCNN module were kept fixed at $\{C_1, C_2, \ldots, C_5\} = \{25, 25, 25, 50, 50\}$. For all the models reported, we compute the area under the curve (AUC) score. The AUC score for all the models were calculated on around 30K query-passage (Q-P) pairs out of our total dataset of 200K Q-P pairs.

Model	P@1
BiDAF	0.58
R-net	0.47
AQuPR	0.788
AQuPR + Op	**0.877**

Table 3: Experiment results comparing retrieval performance.

For Web queries, P@1 beats other P@k metrics since the additional effort to find the passage from the link is only invested for the top link. We report P@1 results in Table. 3.

One interesting observation from the results obtained is the fact that although BiDAF and R-net based attention layer produce comparable AUC scores compared to our model, they fail to obtain a good P@1 score as shown in Table. 3. This can be mainly attributed to the design differences inherent to the different models. BiDAF and R-net based models were designed to identify the correct *span* from the positive passage for a given query. This necessitated additional focus from attention layer to the passage words. On the other hand,

delete a commit from git To remove the last commit from git, you can simply run git reset –hard HEAD^If you are removing multiple commits from the top, you can run git reset –hard HEAD~2 to remove the last two commits. . . .
how to animate single letters in powerpoint In the Custom Animation task pane, click the drop down arrow beside the new animation. Select Effect Options.... In the dialog box that appears, the Effect tab should be selected. Click the drop down arrow beside Animate text. Choose either By word or By letter. . . .
create shape in visio 2013 Click the Developer tab, then click the rectangle shape. Click in the drawing area. A rectangle will appear. This is the starting point of your drawing. Click a corner of the rectangle and drag it into the shape you want. Click in the drawing area again and create a second rectangle using Steps 1-3. . . .
laptop suddenly shuts off power Computers come with heat sinks and fans to keep the CPU cool because if this component overheats the computer will shut down so that the CPU isn't damaged any further. . . .

Table 4: Positive results found by our method. Query, followed by response.

minimize all windows in mac Longtime Mac users should be familiar with this feature which lets you double-click anywhere in the titlebar to minimize a window. To do this in OS X, you'll have to enable the feature in preferences . . . **[Shows Result of Minimizing one window]**
does not point to a valid jvm install You can try this Android Studio error: "Environment variable does not point to a valid JVM instalation". #N#this works for me. If you start 64bit Android Studio, you have to add JAVA_HOME as . . . **[passage contains a link to the correct answer]**
how is git better than svn SVN supports exclusive access control svn lock which is useful for hard-to-merge files. SVN supports binary files and large files more easily (and doesn't require copying old versions everywhere). Adding a commit involves considerably fewer steps since there isn't any pull/push and your local changes are always implicitly rebased on svn update. . . . **[answer reverses the polarity]**

Table 5: Negative results found by our method. Possible explanations are shown in brackets after the passage.

our model was designed to identify the best passage based on the query, and hence, equal weight needs to be assigned to both query and passage.

Some of the positive results obtained from our method are shown in Table. 4. The positive results are exact matches to the human judged top answers. The negative results shown in Fig. 5 show some of the cases where the method fails to find the best answer. Plausible explanation for the failures are also mentioned along with the queries. Due to extensive dependence on word embeddings, these models perform well on semantic similarities amongst entities, which are usually captured well by word embeddings. However, web queries can be very peculiar in nature and many times there are implicit mentions to virtually everything other than nouns (including verbs, adjectives, prepositions, conjunctions and wh-determiner). As a result, it becomes increasingly difficult for the biLSTM layer to learn the syntactic structure of the query and then map it back to the passage. Consequently, the model learns to focus exclusively on *remembering all the nouns* correctly and weights on attention also focus mainly on matching nouns correctly, which results in a lot of intent mismatch based on syntactic structure of the query.

3 CONCLUSION

In this paper we propose an attention based bi-LSTM model for query answering for technology oriented searches. We propose a principled technique for collecting query passage pairs from the Web. We start with a large pool of queries and then systematically filter it down to generate a very focused query set. Our method provides a unique flow to the query passage pairs where they share equal importance in the final representation learned through the biLSTM modules. We introduce a simplified yet elegant attention mechanism which competes extremely well against more complicated models proposed in the recent literature. Our results compare well with the state-of-the-art techniques in this field. In this paper we have proposed an answering based retrieval system for a niche domain, but would like to take this work to newer domains as well as newer research directions.

REFERENCES

[1] Pranava Swaroop Madhyastha, Mohit Bansal, Kevin Gimpel, and Karen Livescu. 2015. Mapping Unseen Words to Task-Trained Embedding Spaces. *CoRR* abs/1510.02387 (2015). http://arxiv.org/abs/1510.02387

[2] Tri Nguyen, Mir Rosenberg, Xia Song, Jianfeng Gao, Saurabh Tiwary, Rangan Majumder, and Li Deng. 2016. MS MARCO: A Human Generated MAchine Reading COmprehension Dataset. *arXiv* (November 2016).

[3] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. GloVe: Global Vectors for Word Representation. In *EMNLP*. 1532–1543.

[4] Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ Questions for Machine Comprehension of Text. *CoRR* abs/1606.05250 (2016). http://arxiv.org/abs/1606.05250

[5] Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional Attention Flow for Machine Comprehension. *CoRR* abs/1611.01603 (2016). http://arxiv.org/abs/1611.01603

[6] Rupesh K Srivastava, Klaus Greff, and Juergen Schmidhuber. 2015. Training Very Deep Networks. In *NIPS*. 2377–2385.

[7] Shuohang Wang, Mo Yu, Jing Jiang, Wei Zhang, Xiaoxiao Guo, Shiyu Chang, Zhiguo Wang, Tim Klinger, Gerald Tesauro, and Murray Campbell. 2017. Evidence Aggregation for Answer Re-Ranking in Open-Domain Question Answering. *CoRR* abs/1711.05116 (2017).

[8] Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated Self-Matching Networks for Reading Comprehension and Question Answering. In *ACL*.

[9] Jason Weston, Sumit Chopra, and Antoine Bordes. 2014. Memory Networks. *CoRR* abs/1410.3916 (2014). arXiv:1410.3916 http://arxiv.org/abs/1410.3916

[10] Yi Yang, Scott Wen-tau Yih, and Chris Meek. 2015. WikiQA: A Challenge Dataset for Open-Domain Question Answering. *TACL* (September 2015).

[11] Wenpeng Yin, Hinrich Schütze, Bing Xiang, and Bowen Zhou. 2016. ABCNN: Attention-Based Convolutional Neural Network for Modeling Sentence Pairs. *TACL* 4 (2016), 259–272.