Three-stage question answering system with sentence ranking

Daria Soboleva[1] and Konstantin Vorontsov[2]

[1] Moscow State University, Moscow, Russia
daria_soboleva_vmk@mail.ru
[2] Moscow Institute of Physics and Technology, Moscow, Russia
k.v.vorontsov@phystech.edu

Abstract. We explore a recently proposed question answering system. We developed a high speed modification based on dividing the question answering system into three consecutive stages. The first step is to find the candidate documents that most likely contain the answer to the question. The second step is to rank sentences by the probability of having a correct answer to the question. The third step is to find the exact phrase that answers the question. At the third step we used a recently proposed recurrent bidirectional neural network predicting the beginning and the end of a response. In this paper we showed that the proposed question answering system allows to speed up its work without significant losses in the quality. For each step we also explored the feature space construction techniques allowing to improve the final quality.

Keywords: question answering · ranking system · document retrieval· recurrent neural network

1 Introduction

This paper considers the problem of question answering systems. The question answering system is the information system capable of accepting questions and answers to the questions in a natural language. "When Elizabeth the second was born?" (running example) and expects an answer as in word/phrases or a sentence. An answer example: "in 1926".

In this paper we developed the question answering system on the collection of text documents from Russian Wikipedia. Wikipedia is an open multi-lingual universal Internet encyclopedia that is a unique source of knowledge. Wikipedia texts are supposed primarily for person reading. That allows to use them for language modeling.

In recently proposed question answering system [1] the authors explored two consecutive steps to build it. The first step was the document retrieval allowing to find k most relevant documents for a particular question. The second step was the document reading where the machine reading model was proposed. This model was able to find the correct answer to the question among k relevant documents received at the first step. The authors supposed the right answer to

be a phrase in the relevant document. At the second step they developed a model predicting for each pair of the question and the document the beginning and end of the answer in the text of this document. This approach is general and can be used to build a question answering system on any collection of text documents.

We proposed a similar question answering system for Russian language collection of text documents adding the third step for ranking the sentences in the k most relevant documents. The third step allowed us to accelerate the model implementation phase without losing much quality.

We also explored the features space construction techniques based on lexical and semantic similarity between the question and the document at the second and the third step. These features improved the third step model by 10–15% performing on the second step closely to the 100% quality.

2 Related Work

In the early 1960 the question answering system was the search engine finding an answer to the question in the collection of unstructured data. Scientists have suggested that computers should help people answer the questions using natural language. At that time the question answering system was a set of simple rules [11]. Today the question answering system does not fit into the set of the simple rules due to the active growth of information technologies and knowledge bases.

There are large series of studies of question answering systems at the TREC conference. In paper [12] authors proposed to build a question answering system with information retrieval techniques. They decided to use a non linear ranking model LambdaMART to rank answer candidates.

Most well-known companies like Facebook, Microsoft, IBM and Google participate in the development of modern question answering systems [1–5].

One of the most recent works was published by Facebook [1]. They proposed a method of question answering system construction based on two successive steps.

The first step was document retrieval finding k most relevant documents for a particular question. This step allows authors to focus on reading only those documents that were likely to be relevant to the question. Document retrieval step used inverted index lookup followed by term vector model scoring. The authors also proposed a method adding n-grams to the search model showed an improvement of the document retrieval step.

The second step was based on the exact phrase searching which is the answer to the question among the k most relevant documents found in the first step. For this purpose the question and the document were encoded using two bidirectional recurrent neural networks [7]. Basic features were Word2Vec the question and the document representations. The authors also considered a set of additional features to encode the document. They used part-of-speech, named entity recognition tags, term frequency features and the set of presence of words from the document in the question indicators.

All features were concatenated into a single vector as the encoding model input. Finally two independent classifiers were constructed to predict the beginning and the end of the answer.

The authors explored the multiple sets of English language datasets like SQuAD [6] and CNN/Daily Mail[2].

The General approach as well as the high quality on the above data allow to use and develop such an approach to the construction of the question answering system in Russian language.

3 Proposed method

In the following we provide the proposed method step-by-step description: (1) Document Retrieval for finding relevant documents, (2) Sentences Ranking to rank sentences in the documents found at the first step and (3) Sentence Reader for extracting answers from a first sentence in the ordered list or from the small set of the first ordered sentences.

3.1 Document Retrieval

Construction the first step model consider a set of classical search algorithms. A set of k relevant documents built by the best model will be passed to the next step. We decided to pass $k = 10$ most relevant documents.

Similarly to [1] consider classical models used inverted index lookup followed by term vector model scoring function[3]. Type of scoring function is very important in building such a system. Thus we decided to compare models with different scoring functions. Our first model (TF) determines the relevance between the question and the document by the frequency of intersections between the words in the question and the document. A similar model (TF_unique) uses intersections of unique words from the question and the document as a proximity. Model (TF-IDF) adds to (TF) model an information about the importance of words from the question. The model (BM25) is a modification of (TF-IDF) model with the priori relevance document information addition [8].

Such models often use coincidence not only by words but also by n-grams. In our experiments adding n-grams did not significantly affect the result. The (BM25) model accuracy by questions was close to 100% and we decided not to complicate it more passing the result of it work to the next step.

3.2 Sentences Ranking

Our Sentences Ranking model is used to accelerate the third step model by ranking sentences by the score determined by the relevance between the question and the sentence.

We decided to extract lexical and semantic similarity features between the question and the sentence candidate. Our lexical features were based on the

[3] https://pypi.org/project/Whoosh/

textual coincidence of the question and the sentence words. Semantic features were used to compare the question and the sentence semantics.

We considered the following lexical features: (BM25), (TF-IDF), percentage (TF_%) and number (TF_count) of common words between the sentence and the question.

Our first semantic feature was based on the Word2Vec model [9] pre-trained on the Russian Wikipedia text collection[4]. The feature was the cosine distance between the mean question vector and the mean sentence vector.

We proposed a method constructing a semantic feature based on the question type and the most common named entity recognition tag [10] in the answers to this type of question. We considered a set of different techniques to determine a question type. In the first technique the question type was the interrogative pronoun occurred in the question. Thus we found around 13 different question types (Who, When, Where, How much, Why etc.). The second technique proposes the question type to be a pair of interrogative pronoun and the next word in the question. After filtering by the occurrence of the question type we found a few thousands different question types.

The third and most successful method of determining the question type was based on the clustering model kmeans. TF-IDF question representation was compressed with PCA to 50 dimension. For each of the interrogative pronoun we added the indicators of their presence in the question. Then all features described above were used to construct a clustering model kmeans. As a result the question type was the cluster number. It was enough to use 100 clusters to receive an appropriate result. Each cluster consisted of the close in sense questions. For example one cluster consisted of questions about an event time, the other one – about an event place, etc.

The final feature is the presence in the sentence the named entity tag appropriate to the name entity tag frequently found in the answers to this type of question.

As for machine learning model (ML) we considered a binary classification model predicting the probability and the ranking models. Our best model was binary logistic regression predicting the probability which we used as a rank to sort the sentences. This approach allows us not to lose potentially good sentence candidates at the second step.

The most relevant k documents with the sentences ranked were passed to the next step.

3.3 Sentence Reader

The third step is to find the answer to the question in the sentences arranged in the second step. Similarly to [1] we used the recurrent bidirectional neural network predicting the beginning and the end of the answer in the sentence. We use 3-layer bidirectional LSTMs with $h = 128$ hidden units for both sentence and question encoding, Adamax for optimization and dropout with $p = 0.3$ applied

[4] http://rusvectores.org/ru/models/

to word embeddings and all the hidden units of LSTMs. This architecture was proposed in article [1].

Our basic model was trained on the Word2Vec [9] representations of the question and the tokens in sentences.

In addition to the basic features we used lexical features and semantics features between the question and the token. Our lexical features were BM25 and TF-IDF score between the question and the token.

Similarly to the second step semantic features we added cosine distance between the average question vector an the token vector.

We also added the semantic feature based on question type the same way we did at the third step. The only difference is the final feature now is the set of all named entity tags combinations from the question and the tokens in sentences. We named this features as (Interaction) as it helps the model to understand the interaction between the named entity tag in the question and the token.

In addition followed by [1] for each token we added the named entity tag and a part-of-speech.

We need to say the semantic features used in both the second and the third steps were more suitable for the third step demonstrating stronger improvement. At the third step we considered two algorithms to construct the question answering system. In the first algorithm the Document Reader model predicting the beginning and the end of the answer for all sentences among k documents. In the second algorithm the Sentences Ranking step were used to rank sentences to give the Document Reader the first relevant sentence or a small set of relevant sentences. The Document Reader model predicting the beginning and the end of the answer in the set of arranged sentences. If the probability of the found answer was higher than the threshold then the found answer was given as the final without viewing the remaining sentences. This approach allowed to reduce the time of the question answering system approximately 16 times without significant losses the quality. The best threshold was around 0.25 found on the validation.

4 Data

Our work relies on the data provided by the organizers of the Sberbank Data Science Journey competition[5]. The dataset contained 50K unique question and answer pairs highlighted by assessors in Russian Wikipedia paragraphs. For each question there was a single document containing the correct answer. In the dataset the question in average consisted of 9 words and the answer was about 4 words in average. Wikipedia paragraphs had in average 101 length in words and 7 length in sentences. The questions were presented in natural language and formed by the assessors themselves.

[5] https://github.com/sberbank-ai/data-science-journey-2017

5 Experiments

In the following sections we present evaluations of our Document Retriever, Sentences Ranking and the Document Reader modules. For each module the quality metrics will be described and the final results will be provided.

5.1 Document Retrieval

Table 1 compares models with different scoring functions on the sub-sample contains 1K unique questions. Our quality metric was Accuracy of the first 10 documents. Results indicate (BM25) model to be the most competitive through the all described models demonstrating the Accuracy close to 100%.

Table 1. Comparison of Document Reader models. Sub-sample of 1K unique questions, $k = 10$.

TF	**TF_unique**	**TF-IDF**	**BM25**
0.66	0.76	0.88	**0.97**

5.2 Sentences Ranking

The average by questions Area Under Curve (AUC) was used to demonstrate the ranking quality of Sentences Ranking module. For each particular question we had in average 70 sentences. Results in table 2 indicate that all lexical features can be used separately on the second step because of 94% AUC. That means the right sentence is in average on the 4th place in the arranged list. Finally the best lexical and semantic features were used to construct (ML) model which was a binary logistic classification predicting probability the answer occurs in the sentence. We used this probability as a score function to determine the relevance between the question and the sentence. The (ML) result was 97% AUC guaranteeing the right sentence in average to be on the 3rd place.

Table 2. Comparison of Sentences Ranking models on the validation set.

TF-IDF	**BM25**	**Term_%**	**Term_count**	**ML**
0.94	0.94	0.94	0.93	**0.97**

5.3 Sentence Reader

The Sentence Reader module was the final step in our question answering system. We used accuracy of the beginning (start), the end (end) and complete coincidence of the answer determination (exact). Table 3 provides the results on the 10th epoch. We can see that each individual feature improves the quality of the basic model (Base). The greatest improvement shows the (Interaction) feature which was based on the question type and the most frequent named entity tag in the answers to this question type. The final model used all the features described above demonstrating the best accuracy of determining the beginning, the end and the exact match between predicting answers and the right answers.

The Figure 1 shows arranged model with threshold around 0.25 is a competitive one accelerating not arranged model around 16 times on 100 questions from the validation set.

The Figure 2 shows the dependence between model quality, execution time and number of sentences viewed. We demonstrated that the quality between 3 and 5 sentences viewed differs by 1% but the execution time is already 5 times different.

We also considered the results on the 40th epoch for the best model. Accuracy of determination both answer ends was around 60%. Full match accuracy was around 50% on both training and validation sets.

Table 3. Comparison of Sentence Reader models on the validation set. The 10 epoch.

	Base	**Ner, Pos**	**Cosine_dist**	**BM25, TF-IDF**	**Interaction**	**ALL**
start	45.57	49.36	50.16	50.88	51.31	**52.63**
end	42.25	47.03	47.72	47.89	49.76	**50.14**
exact	29.01	33.71	34.58	34.90	36.38	**36.88**

6 Conclusion

In this paper we studied a task to create a question answering system for Russian language. We proposed a technique to accelerate the recently proposed methods by separating the question answering system into the three successive steps. This technique allowed us to reduce the time of the question answering system approximately 16 times without significant losses the quality. At the first step Document Reader step was used to find the set of the k most relevant documents. The best model was BM25 demonstrating the quality close to 100%. The second Sentences Ranking step arranged the sentences in the set of k most relevant documents. The best model was binary logistic regression predicting the probability which we interpreted as a score function. This model was trained on the set of best lexical and semantic features. The final Sentence Reader step found

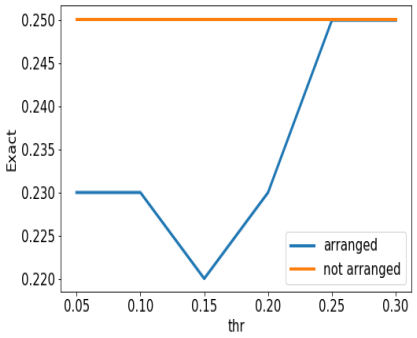

Fig 1. Quality by threshold. Subsample 100 questions.

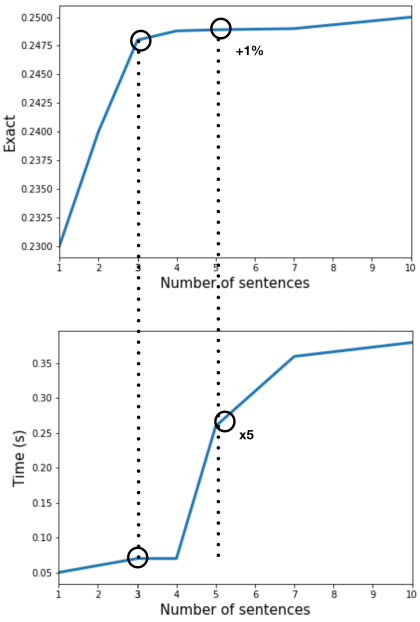

Fig. 2.: Quality and time by number of sentences viewed. Subsample 100 questions.

the answer in the arranged list of sentences. The best model was a recurrent bidirectional neural network trained on the set of basic features with addition to interaction, text matching and cosine distance features improving the quality by 15%.

Acknowledgements The research was made possible by Government of the Russian Federation (Agreement 05.Y09.21.0018)

References

1. Danqi Chen, Adam Fisch, Jason Weston and Antoine Bordes. Wikipedia to Answer Open-Domain Questions. // Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers). Vancouver, Canada, 2017, P. 1870–1879.
2. Karl Moritz Hermann, Tom Koisk, Edward Grefenstette, Lasse Espeholt, Will Kay, Mustafa Suleyman, Phil Blunsom. Teaching Machines to Read and Comprehend. //NIPS'15 Proceedings of the 28th International Conference on Neural Information Processing Systems. Montreal, Canada, 2015. P. 1693–1701.
3. David Ferrucci, Eric Brown, Jennifer Chu-Carroll, James Fan, David Gondek, Aditya A Kalyanpur, Adam Lally, J William Murdock, Eric Nyberg, John Prager, et al. Building Watson: An overview of the DeepQA project. //AI magazine, 2010. P.59–79.
4. Huan Sun, Hao Ma, Wen-tau Yih, Chen-Tse Tsai, Jingjing Liu, and Ming-Wei Chang. Open domain question answering via semantic enrichment. //In Proceedings of the 24th International Conference on World Wide Web. ACM. Florence, Italy. P. 1045–1055.
5. Pum-Mo Ryu, Myung-Gil Jang, and Hyun-Ki Kim. Open domain question answering using Wikipedia-based knowledge model. //Information Processing and Management: an International Journal archive Volume 50 Issue 5. NY, USA, 2014. P. 683–692.
6. Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. SQuAD: 100,000+ questions for machine comprehension of text. //Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing (EMNLP).
7. Sepp Hochreiter, Jrgen Schmidhuber. Long Short-Term Memory. //Journal Neural Computation archive Volume 9 Issue 8. MIT Press Cambridge, MA, USA, 1997 P. 1735–1780.
8. S. E. Robertson and S. Walker. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval // Proceedings of the 17th annual international ACM SIGIR conference on Research and develop- ment in information retrieval, 1994, Dublin, Ireland, P. 232–241.
9. Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. //NIPS'13 Proceedings of the 26th International Conference on Neural Information Processing Systems – Volume 2. Lake Tahoe, Nevada, 2013. P. 3111–3119.
10. Korobov M. Morphological Analyzer and Generator for Russian and Ukrainian Languages // Analysis of Images, Social Networks and Texts, 2015, Yekaterinburg, Russia P. 320–332.
11. Phillips, A. V. A question-answering routine. MIT AI Lab. 1960.

12. Denis Savenkov and Eugene Agichtein. Emory University at TREC LiveQA 2016: Combining crowdsourcing and learning-to-rank approaches for real-time complex question answering //Proceedings of the Twenty-Fifth Text REtrieval Conference, Gaithersburg, Maryland, USA, 2016.