Recurrent Networks in AQA Answer Selection

Radoslav Sabol, Marek Medveď, and Aleš Horák

Natural Language Processing Centre
Faculty of Informatics, Masaryk University
Botanická 68a, 602 00, Brno, Czech republic
{xsabol, xmedved1, hales}@fi.muni.cz

Abstract. Unlimited, or open domain, question answering system AQA is being developed and tested with the Simple Question Answering Database (SQAD) for the Czech language. AQA is optimized for work with morphologically rich languages and makes use of syntactic cues provided by the morphosyntactic analysis.
In this paper, we introduce a new answer selection module being developed for the AQA system. The module is based on recurrent neural networks processing the question and answer sentences to derive the most probable answer sentence.
We present the details of the module architecture and offer a detailed evaluation of various hyperparameter setups. The module is trained and tested with 8,500 question-answer pairs using the SQAD v2.1 benchmark dataset.

Keywords: question answering; answer selection; QA dataset; SQAD; AQA

1 Introduction

Open-domain question answering techniques that look for the answer in unstructured textual data often start with identifying the most relevant parts of text, i.e. they select the relevant documents and/or relevant sentences. A success in this answer (sentence) selection procedure is a key predeterminer of the overall accuracy of the technique.

Specifically, given a question and a (large) set of candidate answer sentences (the unstructured textual knowledge base), the task lies in ordering the sentences by a score which reflects the probability of that the correct answer can be extracted from this particular sentence.

The early approaches to answer selection were based on direct exploitation of either syntactic features of the texts [1] or by identifying discourse entities and semantic relations to support the sentence selection process [2]. The recent results are mostly based on machine learning approaches. Starting with the bag-of-words models based on the Textual Entailment problem [3] up to the current state-of-the-art deep neural networks methods [4,5].

In the following text, a new answer selection module of the open domain question answering system AQA [6,5] is presented. Section 2 briefly recaps the

Aleš Horák, Pavel Rychlý, Adam Rambousek (Eds.): Proceedings of Recent Advances in Slavonic Natural Language Processing, RASLAN 2018, pp. 53–62, 2018.

structure of the AQA system. In the next section, the architecture of the new answer selection module exploiting the recurrent neural network approach is detailed with a thorough evaluation in Section 4.

2 The Automatic Question Answering Tool

The Automatic Question Answering tool (AQA [6,5]) represents an open-domain QA system which concentrates on inflected languages that are guided by rich morphological and syntactic systems. AQA takes as a particular example the Czech language.

The AQA system architecture follows a pipeline model which consists of four main parts:

- question processor module
- document selection module
- answer selection module
- answer extraction module

After the input question is asked by a user, the AQA system preprocesses the question and extracts several pieces of information: the question type, the possible answer type, base forms of all words in the question, all phrases contained in the question, and the word tree distances. All these information are used in the following processing levels.

According to the extracted information, the next level selects the most probable document from the AQA textual knowledge base that should contain the answer to the input question. This extraction is based on TF-IDF scoring.

In the next step, the answer selection module goes through each sentence in the selected document and evaluates their similarity score with respect to the input question. This score is computed from multiple similarity features that are implemented in the system such as the tree distance, entity match, or phrase similarity based on word embeddings.

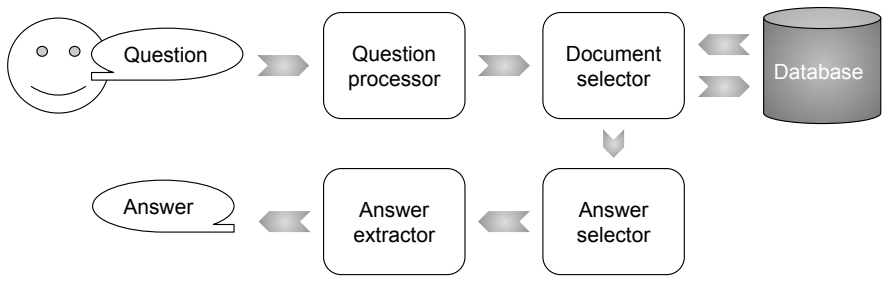


Fig. 1: AQA system work-flow visualization

The last module of answer extraction takes the most probable answer sentence and finds the shortest answer as a sub-phrase of the sentence and provides it as the final answer to the user.

The schema of the AQA system is presented in Figure 1.

The new module introduced in this paper will be employed in the answer selection level as a one of the features that create the scores of candidate sentences.

3 The Answer Selection Architecture

In this section, the architecture of the new answer selection module is detailed. The current implementation is based on the recurrent neural networks (RNN) approach. The general schema takes inspiration from [8], where the authors have introduced a general network schema which jointly learns a similarity measure of two parallel inputs. The schema can be applied to different neural network types, in [8] the convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are evaluated.

The specific RNN architecture of the new answer selection module is presented in Figure 2. Given the question q and pool of candidate answers A, the goal of this network is to rank each input pair with a similarity score $s_\theta(q, a), \forall a \in A$, where the pair with highest ranking is the most probable to be the correct answer.

As an input, the neural network model receives two sequences of word embedding vectors representing the words of the question $q = (q_1, ..., q_L), q_i \in \mathbb{R}^E$ and the words of the candidate answer $a = (a_1, ..., a_M), a_j \in \mathbb{R}^E$ with E being the word embedding dimension. In the first step, the word embedding vectors are independently passed through a bidirectional Gated Recurrent Unit (Bi-GRU) layer consisting of one forward and one backward oriented layer. The Bi-GRU layer creates new hidden representations $Q \in \mathbb{R}^{H \times L}$ and $A \in \mathbb{R}^{H \times M}$ (where H is the hidden output dimension of the Bi-GRU layer), adding contextual information about each token into the new matrix.

The hidden representations of the question and the candidate answer are then combined in the matrix $G \in \mathbb{R}^{L \times M}$ computed as follows:

$$G = \tanh(Q^\intercal \cdot W \cdot A) \tag{1}$$

The central matrix $W \in \mathbb{R}^{H \times H}$ contains learnable weights connecting all elements of the Q and A matrices. The resulting matrix G thus contains soft alignment scores between each token of the question and the answer Bi-GRU outputs. As a projection of this combined information back to the original question and the original candidate answer, the column-wise and row-wise max-pooling followed by the softmax non-linearity is applied to G to obtain attention vectors $g_q \in \mathbb{R}^L$ and $g_a \in \mathbb{R}^M$. The content of g_q can be interpreted as an importance score for each word in q with regard to the candidate answer a and vice versa for g_a.

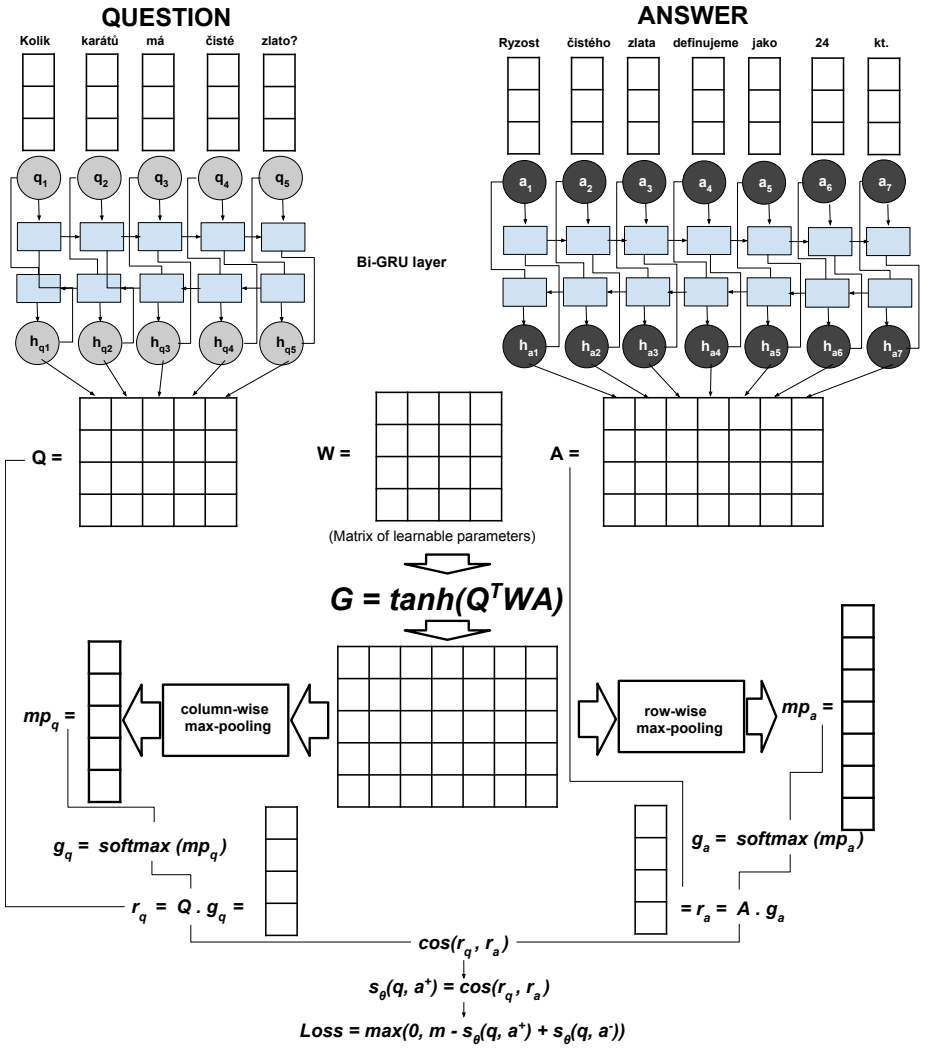
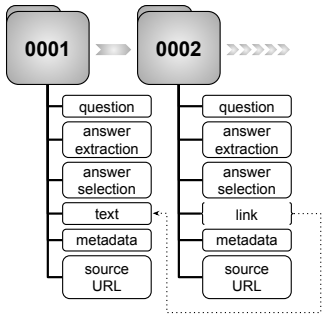


Fig. 2: The RNN architecture of the new answer selection module.

The final representations used for the similarity score computation, $r_q \in \mathbb{R}^H$ and $r_a \in \mathbb{R}^H$ are derived as matrix product of the Bi-GRU hidden outputs Q and A with the corresponding attention vectors. The vectors r_q and r_a thus contain the results of the hidden word representations of the question and the candidate answer weighted by the attentive scores of their mutual relationship:

$$\begin{aligned} r_q &= Q \cdot g_q \\ r_a &= A \cdot g_a \end{aligned} \tag{2}$$



No. of records	8,566
No. of questions	8,566
No. of answer selection	8,566
No. of answer extractions	8,566
No. of unique documents (know.base)	5,417
No. of sentences in KB	433,495
No. of tokens in KB	9,615,966
No. of tokens in questions	69,016
No. of tokens in answer selection	201,069
No. of tokens in answer extraction	19,752

Fig. 3: The SQAD v2.1 file organization and statistics.

The final score of the input pair (q, a) is obtained using the cosine similarity between r_q and q_a.

4 Evaluation

4.1 The Dataset Characteristics

The new answer selection module has been evaluated with the SQAD v2.1 dataset [7]. SQAD is an open source Czech question answering dataset consisting of more than 8,500 question-answer pairs[1] enriched by manually added metadata such as:

- question/answer type labels
- exact answer (answer extraction)
- answer sentence (answer selection)

The exact answer (or answer extraction) is formed by a sub-phrase of the answer selection text. The answer selection sentence comes from identified knowledge base document, but the sentence is too broad to answer the question. Therefore, the answer extraction process is triggered to find the smallest part that can be used as the correct answer to the given question. The database consists of 10 categories of question types and 10 categories of answer types. The distribution matrix of these types is present in Table 1.

To evaluate the module introduced in this paper, the module is trained on a subset of the SQAD database with taking the question and randomly selected sentences from the question source document as negative candidate answers and the answer sentence as the positive candidate answer.

For evaluation purposes the SQAD database has been divided into three parts: training (50% of the dataset), validation (10% of the dataset), and testing

[1] See Figure 3 for the schema of the database content and the current statistics of the SQAD database.

Table 1: SQAD v2.1 distribution matrix of question and answer types

A type / Q type	PER.	DENOT.	ENT.	OTHER	ORG.	DATE /TIME	LOC.	NUM.	ABB.	YES /NO
PERSON	1,016	0	2	0	3	0	2	0	0	0
ENTITY	20	101	1,031	378	204	1	7	1	2	0
ADJ_P.	7	0	8	216	0	0	0	2	0	0
DT./TM.	0	0	1	2	0	1,844	0	4	0	0
LOC.	1	0	14	5	3	0	1,501	0	0	0
NUM.	1	0	0	2	0	0	0	910	0	0
ABBR.	0	1	0	0	0	0	0	0	80	0
CLAUSE	1	0	27	205	6	0	1	1	0	0
VERB_P.	2	0	1	0	0	0	0	0	0	937
OTHER	2	0	1	11	0	0	0	0	0	1

(the remaining 40%). These part are balanced across question/answer type tuples to achieve correct model training and evaluation.

The input word embedding vectors for the question and the candidate answers have been pre-computed by fastText [9] which was trained with large Czech corpus of cleaned web texts (csTenTen17, a corpus of 10 billion words [10]).

4.2 Neural Network Configuration and Training

The Bi-GRU architecture of the new answer selection module was trained and evaluated with the SQAD v2.1 dataset divided into the three subsets – the training set, the validation set, and the testing set. The training set with 4271 question-answer pairs is used to learn the weights and the biases of the model. The validation set of 889 questions provides an unbiased evaluation of the current model parameters after each training epoch. The testing set of 3406 QA pairs is used to evaluate the model after the training is complete.

For each learning epoch, the training set data is shuffled in random order. Each drawn question and the positive answer are randomly supplemented with 50 negative answers sampled from the same text document in the dataset. The input question and each candidate answer are converted to a list of 100-dimensional word2vec embedding vectors. Before each step of the training process, a specific dropout rate is applied on the network input layer.

The final training objective of the answer selection network is defined as a hinge loss [11,12]:

$$Loss = \max\{0,\, m - s_\theta(q, a^+) + s_\theta(q, a^-)\}, \tag{3}$$

where m is a constant margin (0.2 is used as suggested in [8]), s_θ is the cosine similarity as computed by the network with parameters θ, q is the input question and a^+/a^- are the positive/negative answers.

a) one model b) two models at the same time

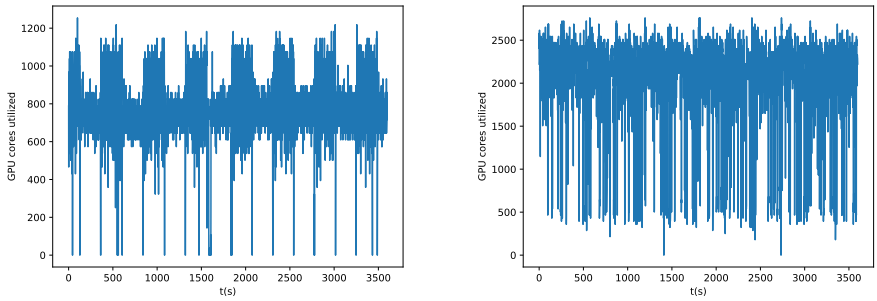


Fig. 4: GPU core utilization for one hour of training. Maximum number of working GPU cores is 3584.

The Stochastic Gradient Descent (SGD) with adaptive learning rate is used as an optimizer. Instead of fixed learning rate, the learning rate λ_t is updated in each epoch t as follows [13]:

$$\lambda_t = \frac{\lambda}{t}, \tag{4}$$

where λ is initial learning rate. The *Loss* is computed for each input of (q, a^*) related to the sampled question, but the network weights are updated only once using the negative answer with the highest score.

The model is trained on 25 epochs, the input data is loaded using multiple workers (on CPU), the training process itself exploits an NVIDIA GeForce GTX 1080 Ti GPU. Figure 4 displays the GPU core utilization for 8 epochs that were executed within 1 hour. The GPU usage noticeably decreases during validation – the reason behind this is the fact that the backpropagation learning algorithm is more demanding than the computations when passing through the validation set. The GPU utilization was 22% on average for training one model, and it raised to 58% (Figure 4 b)) when training 2 models at the same time. The running times for one training epoch were approximately 380 seconds for iterating through the training set, and 260 seconds for the validation set. Although the validation set is smaller in size, all possible answers (sentences in the related document) need to pass through the network for validation, while in the training set only a random sample of 50 candidate answers are used.

4.3 The Results

In this section, the results of models trained with different hyperparameters are presented as shown in Table 2. 27 different models were trained, using 3 values for each of the 3 most influential hyperparameters – the output dimension, the dropout rate, and the initial learning rate.

The output dimension H is the size of the output from the Bi-GRU layer. Note that $H = 2 \cdot h$, where h is the dimension of the hidden vectors of the forward

Table 2: The results for combinations of hyperparameter values

Output size	Dropout	Learning rate λ	Training set (in %)	Validation set (in %)	Test set (in %)
240	0.2	0.05	71.25	54.44	61.77
		0.1	80.05	58.04	65.85
		0.2	85.85	60.29	68.26
	0.5	0.05	67.92	51.74	57.55
		0.1	71.01	53.20	59.95
		0.2	77.68	55.23	63.76
	0.7	0.05	62.83	49.67	54.9
		0.1	47.92	43.31	42.31
		0.2	42.85	36.78	37.17
260	**0.2**	0.05	73.84	54.22	61.89
		0.1	80.84	58.38	65.47
		0.2	**86.09**	61.08	**68.29**
	0.5	0.05	68.15	51.52	58.49
		0.1	69.60	53.99	58.95
		0.2	77.49	55.01	61.13
	0.7	0.05	61.77	50.96	54.9
		0.1	48.74	43.76	43.1
		0.2	44.36	38.69	38.63
280	0.2	0.05	74.59	55.46	62.92
		0.1	80.68	58.72	65.77
		0.2	83.25	**61.52**	68.13
	0.5	0.05	68.83	51.29	58.75
		0.1	69.55	54.11	58.19
		0.2	80.37	56.46	66.18
	0.7	0.05	63.06	50.61	54.67
		0.1	48.68	42.74	42.45
		0.2	43.93	38.92	37.93

or backward GRU layers. These vectors are concatenated to form the output vectors $h_{q_1}, ..., h_{q_L}$ as the columns of the output matrix Q. The output dimension of $H = 260$ has produced the best results, but the other output dimension values of $H = 240$ and $H = 280$ did not substantially degrade the accuracy.

As for the dropout rate, the best results were produced by using the values of 0.2 and 0.5, while 0.7 has decreased the accuracy considerably.

The initial learning rate affects the accuracy in conjunction with the dropout rate: for the dropout of 0.2 and 0.5 increasing the initial learning rate improves the results considerably. On the other hand, the dropout value of 0.7 drops the accuracy by a huge amount independently on the initial learning rate.

For this experiment, the best combination of hyperparameters has been able to find the correct answer in **68.29%**. For a comparison with the previous answer selection module, a different data setup was also evaluated. As the previous module was tested with SQAD v1.0 only, the new module has been here trained with 5265 question-answer pairs from SQAD v2.1 not present in v1.0 and then

tested on the same 3301 QA pairs as the previous module. The new module has reached the accuracy of **66.03%**, which is an improvement of 9.53%.

5 Conclusions and Future Work

We have presented the results of an implementation of a new answer selection module based on the recurrent neural networks approach. The model was trained and evaluated using the SQAD v2.1 question-answering benchmark dataset that consists of 8566 question answer pairs with detailed structured information.

We have provided a thorough evaluation of possible settings of the module hyperparameters with the best attained accuracy of 68.29% when trained on 50% of the dataset and evaluated with 40%, i.e. 3406 questions. In comparison with the previous implementation, the module has reached the accuracy of 66.03% with the SQAD v1.0 data, achieving an increase of almost 10%.

In the next step, we plan to evaluate the whole AQA pipeline accuracy using the newly implemented modules for answer selection and question-answer type detection with the SQAD v2.1 dataset. In an experimental setup, the RNN module will be also tested with sub-sentence phrases instead of full sentences, which would also allow to improve the answer extraction process.

Acknowledgements. This work has been partly supported by the Czech Science Foundation under the project GA18-23891S.

References

1. Litkowski, K.C.: Question-answering using semantic relation triples. In: Proc. of the 8th Text Retrieval Conference (TREC-8). (1999) 349–356
2. Katz, B., Lin, J.: Selectively using relations to improve precision in question answering. In: Proceedings of the workshop on Natural Language Processing for Question Answering (EACL 2003). (2003) 43–50
3. Jijkoun, V., de Rijke, M., et al.: Recognizing textual entailment using lexical similarity. In: Proceedings of the PASCAL Challenges Workshop on Recognising Textual Entailment, Citeseer (2005) 73–76
4. Tay, Y., Tuan, L.A., Hui, S.C.: Multi-cast attention networks for retrieval-based question answering and response prediction (2018)
5. Madabushi, H.T., Lee, M., Barnden, J.: Integrating question classification and deep learning for improved answer selection. In: Proceedings of the 27th International Conference on Computational Linguistics, Association for Computational Linguistics (2018) 3283–3294
6. Medveď, M., Horák, A.: AQA: Automatic Question Answering System for Czech. In: International Conference on Text, Speech, and Dialogue, TSD 2016, Springer (2016) 270–278
7. Medveď, M., Horák, A.: Sentence and word embedding employed in open question-answering. In: Proceedings of the 10th International Conference on Agents and Artificial Intelligence (ICAART 2018), Setúbal, Portugal, SCITEPRESS - Science and Technology Publications (2018) 486–492

8. dos Santos, C.N., Tan, M., Xiang, B., Zhou, B.: Attentive pooling networks. CoRR **abs/1602.03609** (2016)
9. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. CoRR **abs/1607.04606** (2016)
10. Jakubíček, M., Kilgarriff, A., Kovář, V., Rychlỳ, P., Suchomel, V.: The TenTen corpus family. In: 7th International Corpus Linguistics Conference CL2013. (2013) 125–127
11. Weston, J., Chopra, S., Adams, K.: # tagspace: Semantic embeddings from hashtags. In: Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP). (2014) 1822–1827
12. Hu, B., Lu, Z., Li, H., Chen, Q.: Convolutional neural network architectures for matching natural language sentences. In: Advances in neural information processing systems. (2014) 2042–2050
13. Dos Santos, C.N., Zadrozny, B.: Learning character-level representations for part-of-speech tagging. In: Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32. ICML'14, JMLR.org (2014) II–1818–II–1826