Computational Linguistics and Intellectual Technologies:
Proceedings of the International Conference "Dialogue 2023"

Moscow, June 14–17, 2023

Generative Question Answering Systems over Knowledge Graphs and Text

Raushan Turganbay[1]
raushan.turganbay@alumni.nu.edu.kz

Viacheslav Surkov[1]
surokpro2@gmail.com

Dmitry Evseev[1]
dmitrij.euseew@yandex.ru

Mikhail Drobyshevskiy[2]
drobyshevsky@ispras.ru

[1]Neural Networks and Deep Learning Lab, MIPT, Dolgoprodny, Russia
[2]Ivannikov Institute for System Programming of the Russian Academy of Sciences

Abstract

In this paper we describe a generative question answering system which relies on text or knowledge graphs to find supporting evidence. The goal of generative QA is to provide a natural full sentence answer relying on the relevant evidence. Unlike existing models, the system proposed here can generate full answers using knowledge base triplets as evidence and is not restricted to simple questions consisting of one triplet. The generation module is a pretrained encoder-decoder transformer. Additionally, we constructed a new dataset DSberQuAD to train and evaluate the generative QA system in Russian. The new dataset was constructed in a rule-based manner and is an extension of SberQuAD with full sentence answers for each question. The proposed model is a new SOTA for Russian KBQA on RuBQ2.0 dataset. All the code and data from this project are be available on GitHub [1] under Apache license.

Keywords: generative question answering, knowledge base, entity extraction
DOI: 10.28995/2075-7182-2023-22-1112-1126

Генеративные вопросно-ответные системы по графам знаний и тексту

Раушан Турганбай[1]
raushan.turganbay@alumni.nu.edu.kz

Вячеслав Сурков[1]
surokpro2@gmail.com

Дмитрий Евсеев[1]
dmitrij.euseew@yandex.ru

Михаил Дробышевский[2]
drobyshevsky@ispras.ru

[1]Лаборатория нейронных систем и глубокого обучения, МФТИ,
Долгопрудный, Россия
[2]Институт системного программирования им. В.П.Иванникова
Российской академии наук

Аннотация

В данной статье мы описываем генеративную вопросно-ответную систему, которая опирается на текст или графы знаний для поиска подтверждающих доказательств. Целью генеративной вопросно-ответной системы является предоставление ответа на естественном языке в виде полного предложения, основанного на соответствующих доказательствах. В отличие от существующих моделей, предлагаемая здесь система может генерировать полные ответы, используя триплеты тз базы знаний в качестве доказательства, и не ограничивается простыми вопросами, состоящими из одного триплета. Модуль генерации представляет собой предварительно обученный энкодер-декодер. Кроме того, мы предлагаем новый датасет DSberQuAD для обучения и оценки генеративных вопросно-ответных систем на русском. Новый датасет был построен на основе правил и является расширением SberQuAD с полными ответами на каждый вопрос. Предлагаемая модель является новой SOTA для русского на датасете RuBQ2.0. Весь код и данные этого проекта доступны на GitHub [1] под лицензией Apache.

Ключевые слова: генеративная вопросно-ответная система, база знаний, извлечение сущностей

[1]https://github.com/deeppavlov/explainable_qa

1 Introduction

Modern question answering systems have made significant advance in recent years and can find the most relevant answer to question in natural language (Rajpurkar et al., 2016), (Kwiatkowski et al., 2019), (Yang et al., 2019), (Yang et al., 2018). They resort either to knowledge base or unstructured text to find evidence for the answer. Yet, existing models return a short fragment of text as an answer. For example, given a question "What is the capital of France?" the short answer entity "Paris" will be returned, instead of full answer "The capital of France is Paris."

To get full answers, model must be trained to generate a sequence of tokens using relevant facts. One area where generation models are commonly used is goal-oriented dialogue (dialogue goal can be: buy plane tickets, get bus schedules, etc.). In goal-oriented dialog the model should generate a response based on question and the information retrieved from the database. Thus, generating a full answer for factoids can be used in dialog assistants to make the conversation knowledge-grounded and more engaging (Lowe et al., 2015), (Ghazvininejad et al., 2018), (Dinan et al., 2018), (Liu et al., 2018).

This paper proposes QA systems that generate full sentence answers: based on text or knowledge bases (KBs). The model based on text is available for Russian, and models based on KB are available for Russian and English. Also, we propose a generative method on top of the QA system to get full answers. In case of the knowledge-based question answering (KBQA) models, full answer is obtained by feeding the path in the graph to encoder-decoder model trained to generate text from graph. Full answers for text-based model are generated by retrieving paragraphs relevant to the question and using it as input to a different encoder-decoder model that was trained on text data only. The system is built using DeepPavlov library (Burtsev et al., 2018). To train and evaluate more efficiently our text-based generative QA system in Russian, we built DSberQuAD dataset. It extends existing SberQuAD (Efimov et al., 2019) by adding full sentence answers for questions along with short answers.

In summary, we make the following contributions:

(i) We propose a generative question answering model on unstructured text for Russian, which consists of retriever and generation module;

(ii) We extend the traditional KBQA model with answer generation component, which generates full sentence answers from graph triplets;

(iii) We introduce Russian QA dataset with full sentence answers to improve the development of generative QA systems in Russian;

(iv) We introduce a new state-of-the-art on RUBQ2.0 for KBQA task in Russian.

2 Related work

Open domain Question Answering (ODQA) models based on text usually consist of a retriever that finds relevant paragraphs and a component for finding an answer. Relevant paragraphs can be found by calculating dot product between question and paragraph vectors, where the vectors can be sparse representations obtained by methods such as TF-IDF or BM25 (Chen et al., 2017), (Yang et al., 2019). Yet sparse vectorization methods do not perform well when paragraphs are semantically related to the question but do not have any word overlap, thus dense vectorization has been a common choice lately (Karpukhin et al., 2020), (Lee et al., 2019), (Guu et al., 2020). The answer span extraction can be solved using deep learning techniques (Seo et al., 2016), (Wang et al., 2017), (Devlin et al., 2018). These models take a concatenation of question and paragraphs, and assume that answer span is present in one of the paragraphs.

KBQA relies on knowledge graphs to find the correct answer. Knowledge graph is a database that stores information about the world in a structured way. Facts in the knowledge graph can be represented as triplets in `<subject, relation, object>` format. Existing systems use SPARQL queries to answer questions. WQAqua (Diefenbach et al., 2018) and QAmp (Vakulenko et al., 2019) systems start with KB grounding and then construct possible SPARQL queries that return non-empty answers when executed. The most probable queries are determined based on various parameters. On the other hand, NSQA (Kapanipathi et al., 2020) leverages the question's syntactic structure to construct SPARQL

query. These systems do not perform well on complex questions in LC-QuAD and RuBQ2.0, QAmp works over DBpedia only.

To generate full sentence answers several methods were introduced. Answer generation based on text evidence in Tan et al. (2018) is done by adding a decoder on top of answer extraction model, so that the generation model can leverage relevant pieces of evidence. Later, Mitra (2018) expanded on that by adding a pointer-generator network that copies necessary information from relevant paragraph, thus there is no need to extract the answer span first. The task was further improved by using all relevant paragraphs when decoding by Dehghani et al. (2019). Later works approach the problem by using pretrained models. In Fusion-in-Decoder (Izacard and Grave, 2020) relevant documents are concatenated with question and fed into T5 for answer generation. RAG (Lewis et al., 2020), on the other hand, uses BART (Lewis et al., 2019) for generation.

Generative QA based on KBs likewise relies on seq2seq models. GenQA (Yin et al., 2016) retrieves relevant triplets from the graph and used seq2seq models for generation. Later, COREQA (He et al., 2017) extended it by applying a copying mechanism to incorporate relevant facts when generating. However, both systems work only with simple questions.

Here we propose separate generative QA models that rely either on text by generating full answers based on retrieved paragraphs or use KB triplets for generation. We expand on existing work by performing answer generation based on KBs that answers different types of complex questions, including multi-fact, multi-constraint and qualifier-constraint. Moreover, our system is available in Russian (as well as English), which we believe promotes the development of QA models for Russian.

3 Model

Given a question, our model first finds a short answer and then generates full answer using pretrained seq2seq transformers. Since they were pretrained on text only, using them directly to generate answers from KB triplets may not perform well. Thus, we use a model based on JointGT (Ke et al., 2021) which was pretrained jointly on text and graph data and takes into account graph structure. The details of each model's architecture will be outlined further.

3.1 Generative QA over knowledge graphs

The architecture of our KBQA, represented in Figure.1. The model supports Wikidata and DBpedia. To generate full answer, path in the graph from question node to answer node is obtained from SPARQL query and full sentence is generated from that path with a seq2seq model.

3.1.1 SPARQL query template prediction

Since information in KB is stored in a structured way, SPARQL queries are used to extract answers. This component takes a question as input and predicts the most likely SPARQL query from a predefined set of 25 queries. The English model is a BERT classifier fine-tuned on LC-QuAD for 5 epochs with a batch size of 32 and an initial learning rate of $1 \cdot 10^{-5}$.

For Russian model, we use rule-based approach due to lack of training data. The rules can be decomposed to 4 steps: 1) identify the type of answer entity using the relation "instance of"; 2) identify the entity from which the search in the graph will start; 3) answer set is refined using the modifiers in the question; 4) the most likely template is chosen based on the number of entities in the question;

3.1.2 Entity Extraction

Entity extraction converts unstructured text into structured data by finding entities mentioned in question and linking them to their unique identifiers in the graph. First, the entities are identified by classifying each token into an entity or non-entity with RoBERTa token classifier. The model was trained on data obtained from Wikipedia by using anchor texts as entities, as it was done in Ferragina and Scaiella (2011).

The knowledge graph is stored in SQLite database as an inverted index which makes the search process faster and more efficient. Found entities are sorted in ascending order of Levenshtein distance between

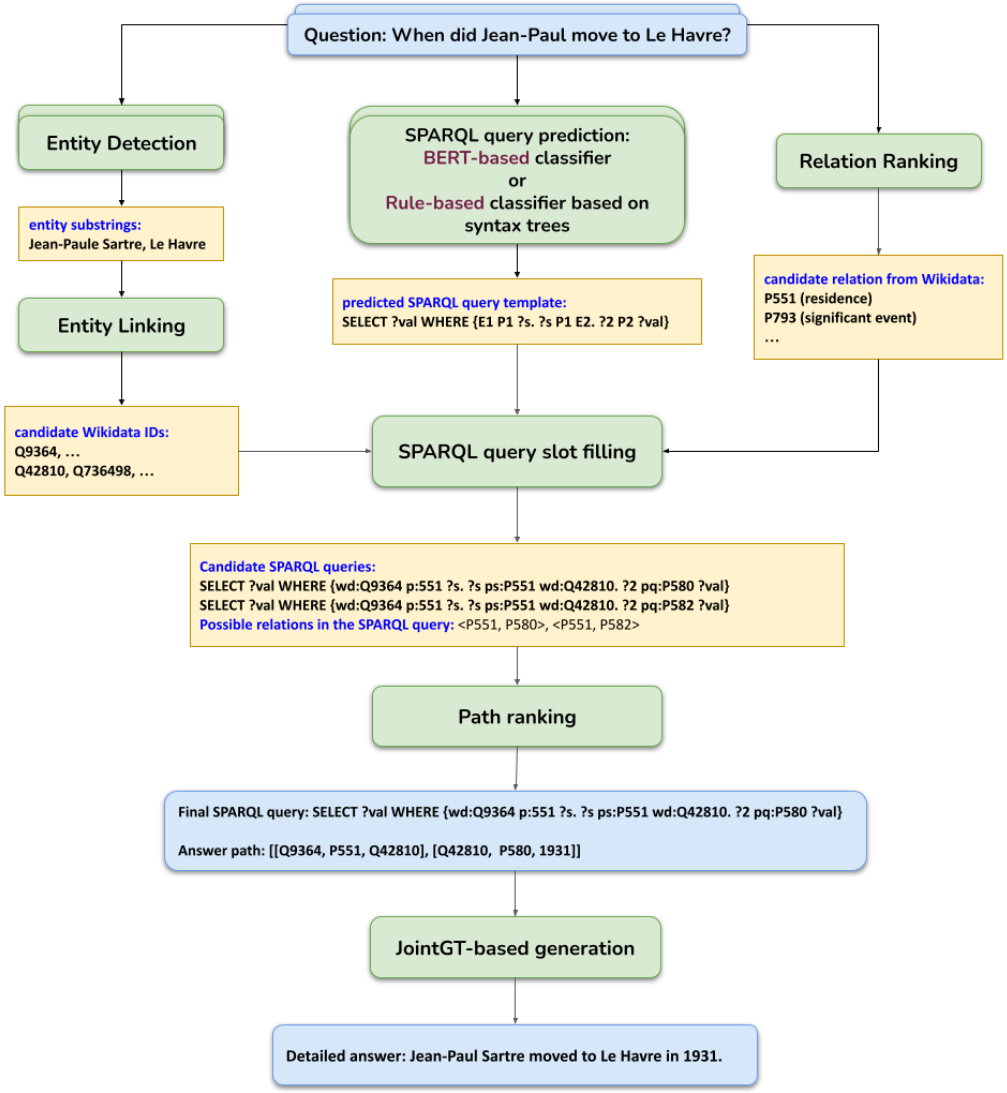


Figure 1: KBQA system

the identified entity substrings and the entity labels from the KB, and further ranked by dot product between the embeddings of the question and entity descriptions.

3.1.3 Relation Ranking

After identifying and linking entities from the question, all relations that are 1-2 hops away from the extracted entities are considered as candidates. They are further ranked using BERT-based model, which receives as input the question and relation separated by SEP token. The final CLS token representation is passed through a feed-forward layer to predict the probability. The model achieved 92% accuracy in the test set. Accuracy was calculated as the percentage of questions where the model predicted the ground truth entity with the highest probability.

3.1.4 Path Ranking

Path Ranking component is designed to determine the most likely combination of relations for SPARQL query by evaluating all possible permutations of candidate relations. It is a BERT-based model that takes as input question and relations separated by SEP token.

The vector of CLS token then goes through a fully connected layer to predict the probability of how well a given set of relations fits into the question. Training for each question uses 1 correct relation (positive sample) and 99 incorrect ones (negative sample). Negative log-likelihood was used as loss

function. The model was trained on LC-QUAD for 3 epochs with batch size of 20 and an initial learning rate of $1 \cdot 10^{-5}$, and reached accuracy=68.2% on validation set.

3.1.5 Full sentence answer generation from knowledge graph triplets

The final output of KBQA is a short answer entity, so we derive a path in the graph from starting entity to answer entity, which we use to generate full answers. The path is represented in triplets. For example, for question `"Which sea is surrounded by Rostock and Kaliningrad?"` the path will be: (("Baltic Sea", "cities", "Rostock"), ("Baltic Sea", "cities", "Kaliningrad")). Generating answers from triplets in graph can enhance our understanding of how the answer was obtained and might be used to improve the interpretability of KBQA models.

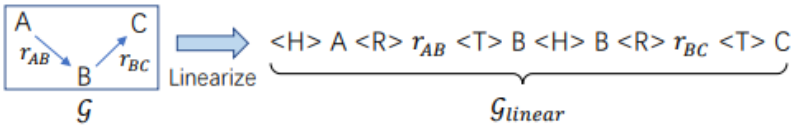


Figure 2: Special tokens <H>, <R> and <T> are used to indicate the head, relations and tail in a triplet.

In this work, the generative component is based on JointGT (Ke et al., 2021), which is a BART-based model with attention between entities and relations in the triplet. First, the path consisting of triplets is linearized as shown in Figure 2. Then, the linearized graph is passed to an encoder with a self-attention layer that captures the interaction between entities and relations in the graph. Finally, encoder output is passed to a decoder to generate an answer.

3.2 Generative QA over text

ODQA system finds answer to the question by searching a large collection of unstructured text, Wikipedia paragraphs in our case. We again use a seq2seq model to generate a full answer, yet the input in this case is a text fragment instead of a path from the graph. The system consists of a retriever, an answer span extractor and generation module, as it can be seen in Figure 3.

3.2.1 Retriever

The retriever component finds top N relevant paragraphs from Wikipedia paragraphs by calculating the scalar product between the vectors of the question and paragraphs. The paragraphs are vectorized using TF-IDF. Each Wikipedia paragraph is tokenized into n-grams (n = 1, 2) and then the hash values of n-grams are calculated using *murmurhash3_32* [2] from sklearn. We get a sparse index where rows correspond to paragraphs and columns to n-gram hash values. The question is vectorized in a similar manner.

The paragraphs are then ranked with BERT classifier to narrow down the list. Each paragraph is classified as relevant or not with certain probability. To train the model, paragraphs that contain an answer were used as positive examples and the rest of the paragraphs as negative examples.

3.2.2 Answer span extraction

To find an answer span from a set of relevant paragraphs, we use a model based on Deeppavlov RuBERT (Kuratov and Arkhipov, 2019) and fine-tuned it on SberQUAD. The model input is a question and paragraph separated by `[SEP]` token. Answer start and end position are predicted by linear transformation of model output. The list of paragraphs where the answer was predicted are used for subsequent answer generation.

3.2.3 Full sentence answer generation from text

For answer generation we experiment with multiple seq2seq backbone transformers. Additionally, we research the effects of transfer learning, by first training the model on MS-MARCO (Nguyen et al., 2016), an extensive English QA dataset. All the seq2seq models were trained for three epochs with

[2] `sklearn.utils.murmurhash3_32`

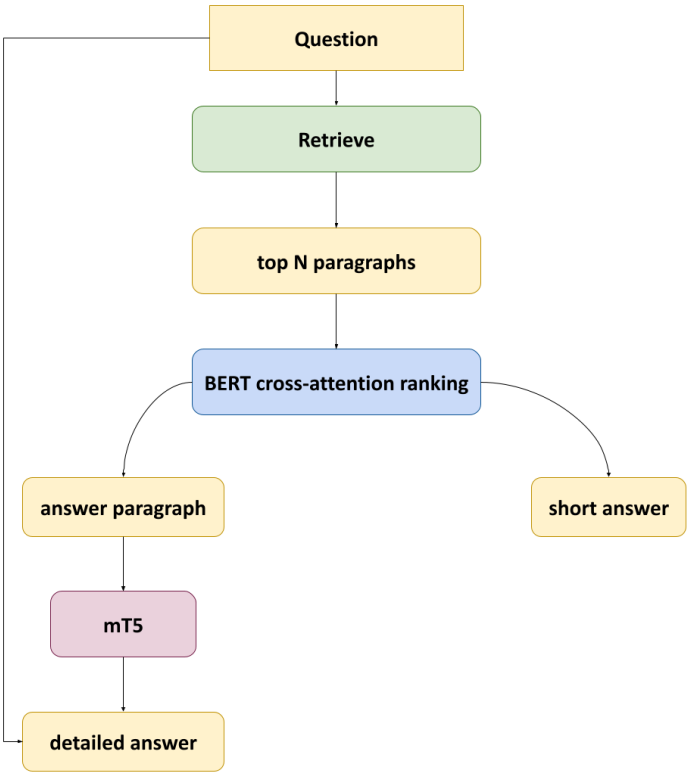


Figure 3: Diagram of the system components. The text-based question answering system finds a short answer, which is further used to generate full answer.

batch size 40 using AdamW optimizer with `learning_rate`=$3 \cdot 10^{-4}$. Below are the three strategies we employed:

1. Multilingual seq2seq model mT5-small (Xue et al., 2021) was fine-tuned on DSberQuAD dataset
2. mT5-small was fine-tuned on DSberQuAD dataset preceded by pretraining on MS MARCO to improve the performance with transfer learning methods.
3. Russian Language model RuT5-base [3] was fine-tuned on DSberQuAD dataset.

Furthermore, we experimented with non-seq2seq methods. We used BERT-based models to extract a short answer substring and further process it with another BERT to generate a long answer. Specifically, we used Deeppavlov RuBERT-base (Kuratov and Arkhipov, 2019), DistilRuBERT and DistilRuBERT-tiny (Kolesnikova et al., 2022) fined-tuned on SberQuAD for answer extraction. The second BERT relies on syntax parser based on Deep Biaffine Attention to construct a long answer on rule-based manner. The details of long answer construction are outlined in Section 4.

4 Datasets

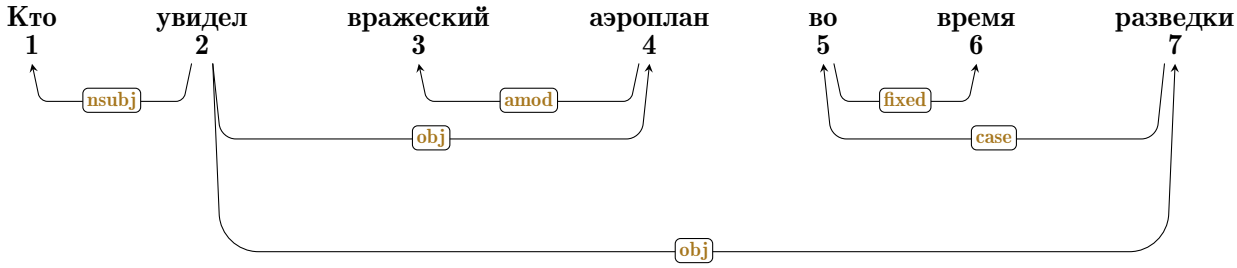
Apart from the model, we present here a new dataset for generative QA which we used for training and evaluation purposes. We hope it will promote the development of generative QA models in Russian. It is a modified version of SberQuAD, we call it DSberQuAD. The dataset creation procedure can be divided into two steps: (1) Replace a question word in the question (e.g. *who, when, why)* with the short substring answer; (2) Form a sentence that follows grammatical agreement rules and change to the `SVO` word order.

For each question and its short answer, their dependency trees are constructed using a syntax parser based on Deep Biaffine Attention (Dozat and Manning, 2017) for building syntax tree and pymorphy2 library (Korobov, 2015) for adjusting cases and numbers. A full answer is constructed based on position and type of question word $qword$ in the text. Totally 10 cases were analyzed, two of which will be

[3] https://huggingface.co/sberbank-ai/ruT5-base

described below. Please refer to Appendix A to see all cases.

In case when $qword$ is a number/case form of $\{кто,\ что\}$, the $qword$ or the whole subtree of the $qword$ is replaced by short answer. In case if the $qword$ or answer are not in nominative case, the sentence is slightly modified to follow case agreement rules. For example, in question "Кто увидел вражеский аэроплан во время разведки?", the dependency tree of which is illustrated below, $qword$ is "Кто" and the answer is "летчики". To construct a long sentence answer, $qword$ is replaced by the answer, and then, since the answer is a plural noun, the verb is changed to its plural form, yielding a full answer sentence —"Летчики увидели вражеский аэроплан во время разведки."



Another case when $qword \in \{когда\}$, it is simply replaced by answer substring. If the answer is a year or percentage, nouns such as "году" or "процентов" are added to the answer. For example, given a question "Когда железная дорога соединила Тильзит и Клайпеду?" and an answer "в 1875", the constructed long answer will be "В 1875 году железная дорога соединила Тильзит и Клайпеду.". It can be noticed, the $qword$ "когда" was replaced by short answer, and since the question is asking for a year, "году" was appended to the answer substring.

Additionally, we manually rewrote answers for a subset of 100 random question from LC-QuAD (Trivedi et al., 2017), which we used to evaluate the performance of the generative model from triplets. For example, given triplets from the KB `[["Google Videos", "developer", "Google"], ["Google Web Toolkit", "author", "Google"]]`, we rewrote the long answer as `"Google Videos and Google Web Toolkit were developed by Google"`.

5 Main Results

Generative model based on text was tested on DSberQuAD validation set. Apart from standard metrics such as BLEU or ROUGE, for evaluation of our model, we introduce *ROUGE-1 lemma* metric. Given as input generated long answer and reference short answer substring, the score is calculated as intersection of short answer tokens in long answer tokens after lemmatizing all words.

The results are presented in Table 1. The highest scoring BERT two-stage approach outperforms RuT5, it is relatively inefficient in terms of the number of parameters. The multilingual T5 performs better when it is additionally pretrained on MS-MARCO, which implies that transfer learning improves the scores (Chizhikova et al., 2023). The best trade-off between the number of parameters and performance is achieved by RuT5 that has the highest ROUGE-1 lemmas score with the least number of parameters used.

Model	Parameters	ppl	ROUGE-1 lemma	SacreBLEU
mT5-small	300M	1.78	48.8	36.8
mT5-small + MS MARCO	300M	1.25	75.9	57.3
ruT5-base	222M	**1.22**	81.0	70.2
BERT-base+BERT-base	180M+180M	—	**88.9**	**87.9**
BERT-base+DistilBert6L	180M+135.4M	—	88.1	82.4
BERT-base+DistilBert2L	180M+107M	—	77.3	64.4

Table 1: Performance metrics of various strategies used for answer generation on DSberQuAD dataset.

Since we have not encountered baselines for generative KBQA on complex questions, we are comparing of our KBQA system without the generative component. We compare our Russian model with

QAmp (Vakulenko et al., 2019), WQAqua (QAnswer) (Diefenbach et al., 2018) and Simba. Table 2 illustrates the evaluation results on RuBQ2.0 (Rybin et al., 2021) dataset, which consists of 2330 questions in test-set. As evaluation metric we used accuracy, which is calculated as the percentage of correct answer entities out of gold answer entities. It can be noted that our KBQA system for Russian outperforms existing approaches, thus achieving SOTA performance on RuBQ2.0. High performance of our KBQA compared to other solutions can be explained by the following:

- We account for the syntax tree when choosing SPARQL query template, which yields higher accuracy, allowing to choose the most possible template;
- We can link entities more accurately since we go beyond term-level matching, and rank entities using the context and entity descriptions;
- To train our system's components we fine-tune a pretrained BERT-base model thus leveraging its language understanding ability;

Question type (quantity)	**DeepPavlov**	**QAnswer**	**SimBa**
questions with answer (1920)	56.0	26.9	25.3
1-hop (1460)	61.1	30.8	32.2
1-hop + reverse (10)	0	0	0
1-hop + count (3)	66.6	33.3	0
1-hop + exclusion (17)	17.7	5.9	0
multi-constraint (304)	50.3	19.7	3.6
multi-hop (55)	1.8	10.9	1.8
qualifier-constraint (22)	45.5	0	4.5
questions without answers (410)	39.8	5.9	86.8
total (2330)	**53.1**	23.2	36.1

Table 2: Comparison of performance metrics on RuBQ2.0 for each question type in the dataset.

For evaluation of our English KBQA system, we use LC-QuAD dataset (Trivedi et al., 2017), which consists of 5000 questions over DBpedia and their respective SPARQL queries. We compare against the following baselines: QAmp, WQAqua and NSQA (Kapanipathi et al., 2020). Performance scores are represented in Table 3. The proposed system surpasses existing baselines on KBQA task. Appendix B shows examples of answers generated by our system.

Model	**Precision**	**Recall**	**F1 score**
QAmp	25.0	50.0	33.0
WQAqua	22.0	38.0	28.0
NSQA	45.0	46.0	44.0
DeepPavlov	**45.0**	**50.0**	**47.0**

Table 3: Performance metrics of KBQA model on LC-QUAD.

To evaluate the generative module, we scored generated full answers against the target by calculating the BLEU score. We used gold triplets as input to the generative component thus evaluating only the generative component. The generative component achieves BLEU-1=70.1 and BLEU-2=56.2 on LC-QuAD. The model's performance can be improved by forcing it to use all entities in the generated text. Thus, method for controllable generation of full answers based on triplets should be examined in future studies.

6 Conclusion and Future Work

In this work, we have described the QA system which is able not only to give a short answer to the question, but also to generate full detailed sentence, thus making the answer more engaging and informative for users. Our system derives answers either from knowledge graph or from unstructured text. In comparison to existing methods (GenQA (Yin et al., 2016), COREQA(He et al., 2017)), our generative

module over KBQA can generate answers to complex questions. For training and evaluation of text-based QA model for long answer generation in Russian, we devised a new dataset DSberQuAD based on SberQuAD. Our system achieves competitive performance on existing QA datasets and the Russian KBQA system achieves SOTA on RuBQ2.0.

One feasible future research direction is to use heterogeneous knowledge sources for generative QA. Although knowledge base approaches excel in addressing complex questions, their effectiveness is frequently hindered by the incomplete nature of the KB. On the other hand, web text consists of numerous facts that are not present in KB, but they are often disorganized and lacking structure. We leave the task of fusing information from both knowledge sources for future work.

References

Mikhail Burtsev, Alexander Seliverstov, Rafael Airapetyan, Mikhail Arkhipov, Dilyara Baymurzina, Nickolay Bushkov, Olga Gureenkova, Taras Khakhulin, Yuri Kuratov, Denis Kuznetsov, Alexey Litinsky, Varvara Logacheva, Alexey Lymar, Valentin Malykh, Maxim Petrov, Vadim Polulyakh, Leonid Pugachev, Alexey Sorokin, Maria Vikhreva, and Marat Zaynutdinov. 2018. DeepPavlov: Open-source library for dialogue systems. *// Proceedings of ACL 2018, System Demonstrations*, P 122–127, Melbourne, Australia, July. Association for Computational Linguistics.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading wikipedia to answer open-domain questions. *arXiv preprint arXiv:1704.00051*.

Anastasia Chizhikova, Vasily Konovalov, and Mikhail Burtsev. 2023. Multilingual case-insensitive named entity recognition. // Boris Kryzhanovsky, Witali Dunin-Barkowski, Vladimir Redko, and Yury Tiumentsev, *Advances in Neural Computation, Machine Learning, and Cognitive Research VI*, P 448–454, Cham. Springer International Publishing.

Mostafa Dehghani, Hosein Azarbonyad, Jaap Kamps, and Maarten de Rijke. 2019. Learning to transform, combine, and reason in open-domain question answering. *// Proceedings of the Twelfth ACM International Conference on Web Search and Data Mining*, P 681–689.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*.

Dennis Diefenbach, Andreas Both, Kamal Deep Singh, and Pierre Maret. 2018. Towards a question answering system over the semantic web. *arXiv preprint arXiv:1803.00832*.

Emily Dinan, Stephen Roller, Kurt Shuster, Angela Fan, Michael Auli, and Jason Weston. 2018. Wizard of wikipedia: Knowledge-powered conversational agents. *arXiv preprint arXiv:1811.01241*.

Timothy Dozat and Christopher D. Manning. 2017. Deep biaffine attention for neural dependency parsing. *arXiv preprint arXiv:1611.01734*.

Pavel Efimov, Leonid Boytsov, and Pavel Braslavski. 2019. Sberquad - russian reading comprehension dataset: Description and analysis. *arXiv preprint arXiv:1912.09723*.

Paolo Ferragina and Ugo Scaiella. 2011. Fast and accurate annotation of short texts with wikipedia pages. *IEEE software*, 29(1):70–75.

Marjan Ghazvininejad, Chris Brockett, Ming-Wei Chang, Bill Dolan, Jianfeng Gao, Wen tau Yih, and Michel Galley. 2018. A knowledge-grounded neural conversation model. *arXiv preprint arXiv:1702.01932*.

Kelvin Guu, Kenton Lee, Zora Tung, Panupong Pasupat, and Ming-Wei Chang. 2020. Realm: Retrieval-augmented language model pre-training. *arXiv preprint arXiv:2002.08909*.

Shizhu He, Cao Liu, Kang Liu, and Jun Zhao. 2017. Generating natural answers by incorporating copying and retrieving mechanisms in sequence-to-sequence learning. *// Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, P 199–208, Vancouver, Canada, July. Association for Computational Linguistics.

Gautier Izacard and Edouard Grave. 2020. Leveraging passage retrieval with generative models for open domain question answering. *arXiv preprint arXiv:2007.01282*.

Pavan Kapanipathi, Ibrahim Abdelaziz, Srinivas Ravishankar, Salim Roukos, Alexander Gray, Ramon Astudillo, Maria Chang, Cristina Cornelio, Saswati Dana, Achille Fokoue, et al. 2020. Leveraging abstract meaning representation for knowledge base question answering. *arXiv preprint arXiv:2012.01707*.

Vladimir Karpukhin, Barlas Oğuz, Sewon Min, Patrick Lewis, Ledell Wu, Sergey Edunov, Danqi Chen, and Wen-tau Yih. 2020. Dense passage retrieval for open-domain question answering. *arXiv preprint arXiv:2004.04906*.

Pei Ke, Haozhe Ji, Yu Ran, Xin Cui, Liwei Wang, Linfeng Song, Xiaoyan Zhu, and Minlie Huang. 2021. Jointgt: Graph-text joint representation learning for text generation from knowledge graphs. *arXiv preprint arXiv:2106.10502*.

Alina Kolesnikova, Yuri Kuratov, Vasily Konovalov, and Mikhail Burtsev. 2022. Knowledge distillation of russian language models with reduction of vocabulary. *1905.07213 arXiv:2205.02340*.

Mikhail Korobov. 2015. Morphological analyzer and generator for russian and ukrainian languages. // Mikhail Yu. Khachay, Natalia Konstantinova, Alexander Panchenko, Dmitry I. Ignatov, and Valeri G. Labunets, *Analysis of Images, Social Networks and Texts*, volume 542 of *Communications in Computer and Information Science*, P 320–332. Springer International Publishing.

Yuri Kuratov and Mikhail Arkhipov. 2019. Adaptation of deep bidirectional multilingual transformers for russian language. *arXiv preprint arXiv:1905.07213*.

Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, et al. 2019. Natural questions: a benchmark for question answering research. *Transactions of the Association for Computational Linguistics*, 7:453–466.

Kenton Lee, Ming-Wei Chang, and Kristina Toutanova. 2019. Latent retrieval for weakly supervised open domain question answering. // *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics*, P 6086–6096, Florence, Italy, July. Association for Computational Linguistics.

Mike Lewis, Yinhan Liu, Naman Goyal, Marjan Ghazvininejad, Abdelrahman Mohamed, Omer Levy, Ves Stoyanov, and Luke Zettlemoyer. 2019. Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension. *arXiv preprint arXiv:1910.13461*.

Patrick Lewis, Ethan Perez, Aleksandra Piktus, Fabio Petroni, Vladimir Karpukhin, Naman Goyal, Heinrich Küttler, Mike Lewis, Wen-tau Yih, Tim Rocktäschel, et al. 2020. Retrieval-augmented generation for knowledge-intensive nlp tasks. *arXiv preprint arXiv:2005.11401*.

Shuman Liu, Hongshen Chen, Zhaochun Ren, Yang Feng, Qun Liu, and Dawei Yin. 2018. Knowledge diffusion for neural dialogue generation. // *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, P 1489–1498.

Ryan Thomas Lowe, Nissan Pow, Laurent Charlin, and Joelle Pineau. 2015. Incorporating unstructured textual knowledge sources into neural dialogue systems.

Rajarshee Mitra. 2018. A generative approach to question answering. *arXiv preprint arXiv:1711.06238*.

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 preprint arXiv:1611.09268*.

Pranav Rajpurkar, Jian Zhang, Konstantin Lopyrev, and Percy Liang. 2016. SQuAD: 100,000+ questions for machine comprehension of text. // *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*, P 2383–2392, Austin, Texas, November. Association for Computational Linguistics.

Ivan Rybin, Vladislav Korablinov, Pavel Efimov, and Pavel Braslavski. 2021. Rubq 2.0: An innovated russian question answering dataset. // Ruben Verborgh, Katja Hose, Heiko Paulheim, Pierre-Antoine Champin, Maria Maleshkova, Oscar Corcho, Petar Ristoski, and Mehwish Alam, *The Semantic Web*, P 532–547, Cham. Springer International Publishing.

Minjoon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv preprint arXiv:1611.01603*.

Chuanqi Tan, Furu Wei, Nan Yang, Bowen Du, Weifeng Lv, and Ming Zhou. 2018. S-net: From answer extraction to answer generation for machine reading comprehension. *arXiv preprint arXiv:1706.04815*.

Priyansh Trivedi, Gaurav Maheshwari, Mohnish Dubey, and Jens Lehmann. 2017. Lc-quad: A corpus for complex question answering over knowledge graphs. // *International Workshop on the Semantic Web*.

Svitlana Vakulenko, Javier David Fernandez Garcia, Axel Polleres, Maarten de Rijke, and Michael Cochez. 2019. Message passing for complex question answering over knowledge graphs. *// Proceedings of the 28th acm international conference on information and knowledge management*, P 1431–1440.

Wenhui Wang, Nan Yang, Furu Wei, Baobao Chang, and Ming Zhou. 2017. Gated self-matching networks for reading comprehension and question answering. *// Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, P 189–198.

Linting Xue, Noah Constant, Adam Roberts, Mihir Kale, Rami Al-Rfou, Aditya Siddhant, Aditya Barua, and Colin Raffel. 2021. mT5: A massively multilingual pre-trained text-to-text transformer. *// Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, P 483–498, Online, June. Association for Computational Linguistics.

Zhilin Yang, Peng Qi, Saizheng Zhang, Yoshua Bengio, William W Cohen, Ruslan Salakhutdinov, and Christopher D Manning. 2018. Hotpotqa: A dataset for diverse, explainable multi-hop question answering. *arXiv preprint arXiv:1809.09600*.

Wei Yang, Yuqing Xie, Aileen Lin, Xingyu Li, Luchen Tan, Kun Xiong, Ming Li, and Jimmy Lin. 2019. End-to-end open-domain question answering with bertserini. *arXiv preprint arXiv:1902.01718*.

Jun Yin, Xin Jiang, Zhengdong Lu, Lifeng Shang, Hang Li, and Xiaoming Li. 2016. Neural generative question answering. *arXiv preprint arXiv:1512.01337*.

Appendix A Cases analyzed when constructing the DSberQuAD dataset

Below are presented the 10 cases we analyzed when constructing the DSberQuAD dataset. If the question does not fit into any of the cases, the answer is left without changes.

Case 1 When the *qword* ∈ *{кто, что}* and it is the root of the syntactic tree, the *qword* (and sometime the whole subtree of the qword is replaced bye the answers *a*. Also, a hypthen is added. (Table 1).

Q	Кто такой Пушкин?
A	Великий русский поэт
LA	Пушкин - это великий русский поэт.

Table 1: Case 1 for constructing DSberQuAD.

Case 2 *qword* — If it is the case form of the *qword кто, что* , the *qword* (and sometime the whole subtree of the *qword*) is replaced by the answer. The answer's case and number is changed to align with the case and number of the *qword* and the necessary prepositions are added. If the *qword* is not the subject of the question, then the answer or the whole subtree is place after the predicate (Table 2).

Q	Чем заболел Байрон в Миссолонги?
A	лихорадкой
LA	Байрон заболел лихорадкой в Миссолонги.
Q	Кто увидел вражеский аэроплан во время разведки?
A	летчики
LA	Лётчики увидели вражеский аэроплан во время разведки.
Q	Что было открыто во дворце после восстановления?
A	Национальная галерея
LA	Национальная галерея была открыта во дворце после восстановления.

Table 2: Case 2 for constructing DSberQuAD.

Case 3 If the *qword* is the case form of *какой, чей*, depending on whether the answer is a noun phrase, an adjective or a date, the *qword* (or the hole subtree) is replaces with the answer. Case and number of the answer is changed according the agreement rules, necessary prepositions are added. The sentence order is changes to *Subject-Verb-Object*. Additional post-processsing was done on questions that start with *"Какое название/наименование..."; "В каком году..."; "Чей..."* (Table 3).

Q	В каком году отец Оригена был убит?
A	В 202 году
LA	Отец Оригена был убит в 202 году.
Q	В каком замке был заключен Дидро?
A	Венсенском
LA	В венсенском замке был заключен Дидро.
Q	Чьи ротовые аппараты разнообразны?
A	Чешуекрылых
LA	Ротовые аппараты чешуекрылых разнообразны.

Table 3: Case 3 for constructing DSberQuAD.

Case 4 If the *qword* is a case form of *каков*, the *qword* is replcaed by the answer. The word order remains the same (Table 4).

Q	Какова температура кипения воды?
A	100 градусов
LA	100 градусов — температура кипения воды.

Table 4: Case 4 for constructing DSberQuAD.

Case 5 When $qword$ = *когда*, the $qword$ is replaced by the answer. The word order remains the same. Additional post-processing is added if the answer is a year. In that case the answer is changed to have the format *"В X году"* (Table 5).

Q	Когда мосты стали строить из железобетона?
A	В XX веке.
LA	В XX веке мосты стали строить из железобетона.
Q	Когда железная дорога соединила Тильзит и Клайпеду?
A	в 1875
LA	В 1875 году железная дорога соединила Тильзит и Клайпеду.

Table 5: Case 5 for constructing DSberQuAD.

Case 6 When $qword \in$ *{где, куда, откуда, докуда}* , the $qword$ =is replaced by the answer. The word order is changed to *Subject-Verb-Object* (Table 6).

Q	Куда переехала Фанни из Сан-Франциско?
A	в Монтерей
LA	Фанни переехала в Монтерей из Сан-Франциско.

Table 6: Case 6 for constructing DSberQuAD.

Case 7 When $qword \in$ *{почему, отчего}* , answers are not rewritten in full forms since the original asnwers are already full sentences.

Case 8 When $qword$ = *как*, the root of the T_Q is placed at the end of the sentence. The question word is removed and the answer is appended to the end of sentence. If question starts with *"Как быстро... "Как долго..."*, these are assumed to be the $qword$ and removed all together. If the $qword$ is *"Как переводится... "Как описывается"* or the answer is a noun in accusative case, then a word *так* is added to the answer (Table 7).

Q	Как отводятся излишки тепла у млекопитающих?
A	потоотделением
LA	Излишки тепла у млекопитающих отводятся потоотделением.
Q	Как переводится слово каллиграфия?
A	Путь письма.
LA	Слово каллиграфия переводится как путь письма.
Q	Как быстро протекает митоз?
A	1—2 часа.
LA	Митоз протекает 1—2 часа.

Table 7: Case 8 for constructing DSberQuAD.

Case 9 When $qword$ = *сколько*, the $qword$ is replaced by the answer. To make the sentence grammatically correct, additional post-processing is done if the question asks for the percentage or degree values. Prepositions are added or deleted and the case/number forms are changed to follow the agreement rules (Table 8).

Q	При скольки градусах плавится альфа-цирконий?
A	1855 °C
LA	При 1855 °C плавится альфа-цирконий.
Q	Сколько астероидов обнаружено в настоящий момент в солнечной системе?
A	Сотни тысяч астероидов.
LA	Сотни тысяч астероидов обнаружены в настоящий момент в солнечной системе.
Q	Сколько человек родилось в 2008 году у граждан Швейцарии?
A	1147 человек
LA	1147 человек родилось в 2008 году у граждан Швейцарии.

Table 8: Case 9 for constructing DSberQuAD.

Case 10 When $qword = ли$, the $qword$ is removed and its parent p is replaced by the answer.The word order is changed to *Subject-Verb-Object* (Table 9).

Q	Разная ли величина приемлемого риска для различных обществ?
A	различная
LA	Величина приемлемого риска для различных обществ различная.
Q	Гигроскопичен ли германат висмута?
A	абсолютно негигроскопичен
LA	Германат висмута абсолютно негигроскопичен.

Table 9: Case 10 for constructing DSberQuAD.

Appendix B Examples of answers by our English KBQA system in comparison to the compared baselines.

Here we illustrate examples of how our system generates full sentence answers to questions using path triplets in the graph. As i was mentioned, using path triplets for generation can enhance the interpretability of the KBQA system by demonstrating how complex questions were broken into multiple triplets. Table 1 shows some examples of questions from LC-QuAD dataset, short answer entity found by out system, path triplets in (subject, relation, object) format and the generated full answer.

Question	**Answer**	**Triplets**	**Generated Text**
Is Essex the Ceremonial County of West Tilbury?	Yes	<'West Tilbury', 'Ceremonial County', 'Essex'>	West Tilbury is in the County of Essex.
What nation is Hornito located in, where Jamie Bateman Cayn died too?	Panama	<'Jaime Bateman Cayón', 'death place', 'Panama'>, <'Hornito, Chiriquí', 'country', 'Panama'>	The death place of Jaime Bateman Cayón is Hornito, Chiriquí, Panama.
Who are the shareholder of the soccer club for whom Steve Holland plays?	Roman Abramovich	<Steve Holland', 'current club', 'Chelsea F.C.'>, <'Chelsea F.C.', 'owner', 'Roman Abramovich'>	Steve Holland's current club is Chelsea F.C. which is owned by Roman Abramovich.
Who is the chancellor of Falmouth University?	Dawn French	<'Falmouth University', 'chancellor', 'Dawn French'>	The chancellor of the Falmouth University is Dawn French.
What is the relegious affiliations of Katyayana?	Buddhism	<'Katyayana', 'religion', 'Buddhism'>	The religion of Katyayana is Buddhism.

Table 1: Examples of long answers generated by our system on LC-QuAD dataset.