Reaching out for the Answer: Answer Type Prediction

Kanchan Shivashankar, Khaoula Benmaarouf, and Nadine Steinmetz

Technische Universität Ilmenau, Germany
firstname.lastname@tu-ilmenau.de

Abstract. Natural language is complex and similar statements can be expressed in various ways. Therefore, understanding natural language is an ongoing challenge in the field of Question Answering. To make the process from the question to the answer, the QA pipeline can be broken down to several sub-steps, as e.g. prediction of the potential type of the answer, entity linking and detection of references for properties relevant for the formal query. The SMART Task challenge, co-located with ISWC 2021, focussed on two of these sub-tasks: relation linking and answer type prediction. With this paper, we present our approach for the latter task. Our solution is a two-staged process combining two separate multi-label classification tasks. For the answer category prediction, we utilize the RoBERTa language model. Questions predicted as resource questions are then further classified regarding the concrete answer types – a list of ontology classes – utilizing the BERT language model.

Keywords: Question Answering · Answertype Prediction · Semantic Web

1 Introduction

The discipline of Question Answering(QA) is a very popular research domain. It combines the fields of Information Retrieval (IR) and Natural Language Processing (NLP). The objective is to answer questions posed by humans in natural language (NL) by extracting relevant information from available knowledge bases. Its popularity arises from the increase in the demand to find relevant and useful information from huge amounts of data. This might seem like a very simple problem to begin with. As humans, language is the integral part of our life, we use it in our daily life with ease. However, teaching a computer to understand and interact in NL is very complex. There are several aspects to understanding natural language. It is not only dependent on the syntax and semantics, but we also take in reference of context, tone, body language and other inputs to fully understand. And despite that, we as humans are prone to misinterpretations and misunderstandings. Training a machine to understand and interact

with only written text makes it a real challenge and leaves no gap for mistakes. Additionally, once the text is correctly interpreted, extracting relevant information from the vast amount of information, is like trying to find a needle in a haystack. In general, there are two parts to solving this problem. The first part is to break down the question to extract its meaning. This also includes identifying the expected answer type for the question, which is the problem addressed in this paper. This transformation is performed using different tools from NLP. The next part is to use the extracted meaning to build a query. The query is used to find answers from the data source resp. knowledge base. The extracted answers are then sorted using ranking algorithms to prioritize the most accurate and relevant answer for the question.

The scope of this paper is the prediction of answer types for the question. The data source from which the answer must be retrieved can be structured data such as knowledge bases or unstructured data in the form of documents. In this paper, we are dealing with two popular KBs: Wikidata and DBpedia. We present our solution developed in the context of the SMART challenge (co-located with the ISWC 2021) – specifically Task 1: Answer Type Prediction. We took part in the challenge for both ontologies: Wikidata and DBpedia [13]. Both containing large and distinct answer type classes. The prediction task helps retrieve more accurate and relevant information, but is challenging due to the highly granular (Wikidata: 50k classes and DBpedia: 760 classes) nature of the classes available.

We propose a two-step approach to solve this problem. Firstly, a classification model is used at the higher level to predict the answer category based on RoBERTa. Secondly, to predict the more granular resource types, we use a second multi-label classification based on BERT. The remainder of the paper is organized as following: Section 2 we give an overview on the evolution of the field of Question Answering, KBs and the state of the art methods. Section 3 defines the characteristics of the datasets used. Section 4 introduces our approach. This is followed by a detailed evaluation in Section 5. We conclude with some remarks and scope for future work.

2 Related Work

"Can machines think?" This is one of the most popular questions of the 20th century, posed by renowned Mathematician Alan Turing in his paper "Computing Machinery and Intelligence" [17]. This paper led to the birth of what we today call as Artificial Intelligence (AI). In his paper, the author proposes a simple yet complex problem - the Turing Test. The rules to perform the test however simple, achieving a machine to pass the test is quite complex. The test – simply put – requires a machine to interpret and generate natural language. This is the main objective of all the tasks in Natural Language Processing (NLP). NLP is one of the widely researched fields in AI. From CAPTCHAs, chatbots to the voice activated virtual assistants – all fall under the applications of NLP.

Question Answering is an advanced form of Information Retrieval(IR) systems that help extract concise and short answers from a pool of information to

answer a NL question. There are many architectures to define a QA system but it consists of 3 main parts: question processing, document processing and answer processing. The question processing step involves extracting important information from the question like keywords and answer type. In document processing, a subset of relevant information (for example paragraphs) is selected which could potentially contain the answer. The final stage of answer processing performs detailed analysis of the extracted information from the previous stage in order to locate and extract the answer [11].

Obviously, approaches have evolved over time. The most basic is a linguistic approach, breaking down the question with different text processing techniques such as tokenization, POS tagging and parsing to build a query. Easy access and availability to huge amounts of data, made statistical approaches advantageous. They outperformed traditional linguistic approaches. Another important approach to mention is the pattern matching approach, ideally used in instances which require less computational power with predefined templates or formats for question answering. Both linguistic and statistical approaches are suitable for only closed domain applications [14].

Knowledge Bases(KB) contain information of real world entities as nodes and relations as edges. Each directed edge with head entity and tail entity constitutes a triple also called as a fact. These facts can be accessed through a query language called SPARQL, however, it requires expert knowledge of programming and is difficult to learn for non-programmers. The KBQA systems allow lay person, access to the information, by posing a natural language question. The question is internally converted into a SPARQL query, which retrieves the answer, all without requiring the knowledge of the underlying schema. The underlying process comprises of two main steps namely, Entity (subject or object) Linking and Relation (predicate) Linking. This forms the main pipeline of a KBQA system[2].

Several existing methods for KBQA directly parse the natural language question into a structured query using semantic parsing including syntactic parsing, semantic role labeling etc. Some of them map the natural language question to a triple-based representation, but they fail to capture the original semantic structure of question using triples[7]. Keyword based methods use keywords in the question and map them to predicates by keyword matching. They are successful in answering simple questions and needs to be extended to complex questions. Keyword matching is difficult as it can have many representations in the natural language. This can be solved to an extent by generating alternative queries, but results in many candidate answers to choose from[6].

QA systems can be based on different types of KBs: semantic knowledge graphs (as it is for this paper), relational databases or simple document collections [1]. A knowledge Based Question Answering System (KBQA) is equipped to answer questions on the knowledge stored. With the evolution of KBs and NLP, today the challenge is to answer complex questions with multiple relations amongst multiple subjects. It involves multi-hop reasoning, constrained relations, numerical operations or a combination of the above [4].

Some of the state-of-the-art solutions consist of complex end-to-end neural network models, performing well on various NLP tasks. But they require a lot of computational power, time to train the networks and limit the opportunity to fully understand the underlying problem. The SMART challenge took place for the second time in 2021. The previous edition was co-located with the ISWC 2020 and lists eight participants for the answer type prediction task [12] The best approach used BERT (Bidirectional Encoder Representations from Transformers) and achieved accuracy $> 95\%$ for category prediction and $> 70\%$ accuracy for DBpedia and 68% accuracy for Wikidata resource type predictions [16], proving the performance of neural network models in KBQA systems. But these systems fail to understand the nuances of NL beyond the training data.

Although statistical approaches seem like the best option, the complexity of languages and limit on the computational power requires us to use a combination of both. A hybrid approach combines analytical approaches with statistical approach to build adaptable systems with better performance. The semantic parsing technique aims at representing semantic structures of the questions, and neural networks have shown promising results in KBQA. Combining the semantic parsing and neural network has shown promising results for several KBQA datasets [15] [10].

Answering complex questions using knowledge bases is an open and challenging task. However, there are many challenges to overcome. Multiple predicates are required to constrain and query the answer set. Each KB has a unique underlying ontological structure. The nuances of natural language and complex questions consisting of multiple entity-relation pairs, introduces an additional challenge. There also exists multiple entities in the knowledge base with the same name, which creates ambiguity.

3 Data Analysis

Data analysis is an essential step to get to know the data. It provides insight into the quality and features of the data and helps to decide the type of preprocessing steps to be utilized. In machine learning, quality and quantity of the dataset (train and test) both determine how well the model performs. High quality of the training data improves the models learning curve. Supervised Learning also requires that the data is labeled accurately. Failure to remove or correct noisy data results in wrong predictions. Some of the examples of noisy data are missing or null values, duplicate data, mislabeled data etc. The datasets are based on two different knowledge bases: Wikidata and DBpedia, with different ontologies. Wikipedia is a large collection of structured data, across multiple disciplines and languages. There is a great potential for this data to be searched, analyzed and reused. But this data cannot be easily accessed, queried or downloaded due to lack of management of data. Wikidata aims to overcome such inconsistencies by creating new ways for Wikipedia to manage its data on a global scale [3]. The DBpedia community project also extracts structured, multilingual knowledge from Wikipedia info boxes and tabular summaries and

makes it freely available on the Web using Semantic Web and Linked Data technologies [8]. The structured data can be used to query information from Wikipedia content.

The datasets are provided in json format. Training data fields include

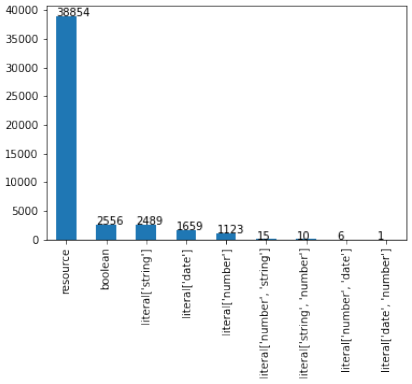
(a) unique ID
(b) question
(c) answer category
(d) answer type

The test datasets only contains fields (a) and (b).

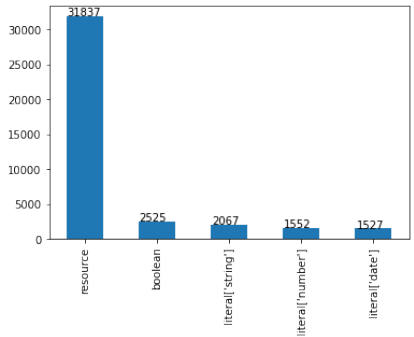
There are 3 different answer categories: `boolean`, `literal`, and `resource`. Literal has three different answer types: `number`, `date` and `string`. The answer type for the category `boolean` is also a boolean. The answer category `resource` requires a list of classes from the respective ontology: 760 classes for DBpedia and 50k classes for Wikidata. We performed some pre-processing and analyses on the datasets. Firstly, to have more data for training the model, we have merged the training datasets of SMART 2020 and SMART 2021 challenges. The combined dataset is checked for `null` and duplicate values and removed from the dataset. Table 1 provides details on the training and test datasets.

Table 1: Statistics on SMART Task Training Datasets.

Knowledge Base	SMART 2020	SMART 2021	Total Training Data	Test Data
Wikidata	18251	43554	46713	10864
DBpedia	17571	36670	39508	10093



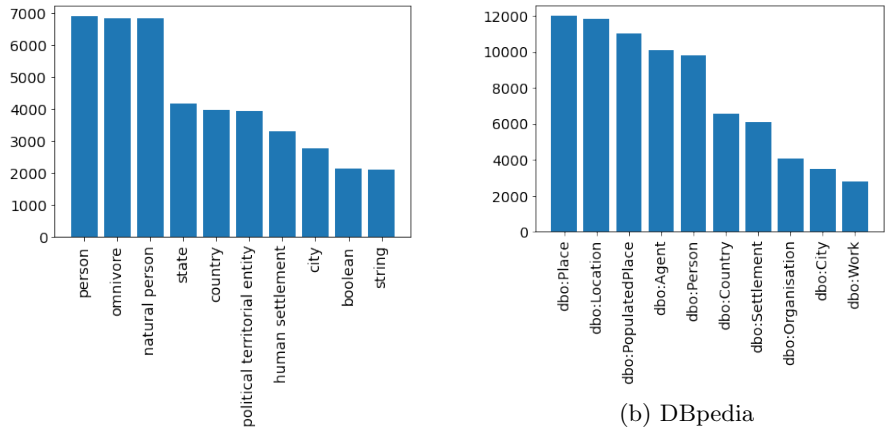
(a) Wikidata



(b) DBpedia

Fig. 1: Category distribution

In addition, we also identified the distribution of different answer categories and types present in the dataset. The most commonly occurring category is `resource` with around 80% of the questions belonging to this category. We also found erroneous labels for category and type fields in the Wikidata training dataset. As shown in Figure 1, there are a few records with multiple different types for the `literal` category, as e.g. `number` and `string` both for the same record.



(a) Wikidata

(b) DBpedia

Fig. 2: Commonly occurring Resource Type labels

The top 10 commonly occurring resource types for both datasets are shown in Figure 2. The Wikidata training dataset contains many questions about persons, and locations on second place. For the DBpedia dataset it is vice-versa. The DBpedia training dataset contains 380 different classes of the DBpedia ontology. 62 classes only occur once in the dataset. For the Wikidata dataset these numbers are 3,311 and 1,027 respectively.

4 Method

Our prediction approach consists of two parts: category prediction and resource type prediction. Both parts are multi-label classification problems, but with different degrees in terms of number of class labels. Figure 3 depicts our overall process.

4.1 Category Prediction

The label categories are split into 5 main classes: `resource`, `literal-number`, `literal-string`, `literal-date`, and `boolean`. Thus, this part is treated as a

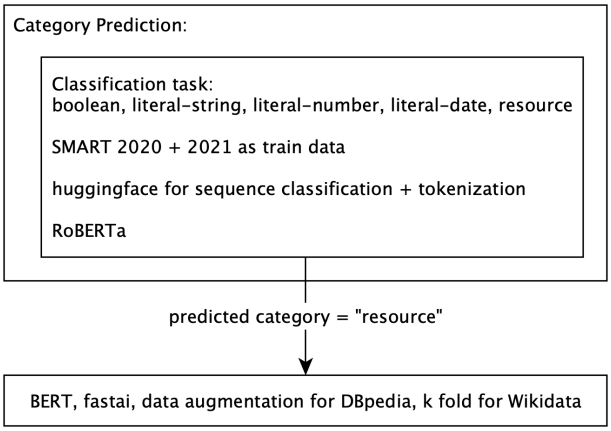


Fig. 3: The overall prediction process in two stages.

classification problem with five different classes. RoBERTa (Robustly optimized BERT Pretraining Approach) is an optimized version of the BERT model with some modification to the key hyper-parameters [9]. We use the roberta-base pre-trained model from the transformers library for both tokenizer and the sequence classification model. Transformers is a huggingface library for NLP, with state of the art general purpose architectures and pre-trained models readily available for use. This model is trained on a large corpus of data in English language and is a case-sensitive model.

Pre-processing BERT can take up to two sentences as input. The second sentence is used in case of a classification task and both sentences are separated by special tokens. The input data for training the classification model should contain the input text, in this case the question and also the expected label i.e, answer category. The model expects a numerical representation of the textual input. This process is called word embedding. First step is the tokenization where the text is split into a sequence of words. Some words are further split into sub-words or characters. This is due to the limitation of words that can be represented in the vocabulary. These tokens are replaced by the indices of the vocabulary. We use a pre-trained tokenizer for our model.

The RoBERTa tokenizer is derived from the GPT-2 tokenizer, using byte-level byte-pair-encoding, which is a form of sub-word tokenization. It has been trained to treat spaces like parts of the tokens, so a sub-word/sub-string that occurs at the beginning of the word (without space) will be encoded differently from the one found at the end of the word (with space). Words not present in the vocabulary are marked as unknown tokens. The batch encoding method of the Roberta tokenizer takes the question in textual format and returns two outputs: input ids and attention masks. The input ids is a list of indices for the

tokens from the vocabulary with separator tokens `CLS` and `SEP` to indicate the beginning and the end of the input to the model. The input questions are also padded to have a uniform length across all input questions. The attention mask consists of sequence of 1s (text) and 0s(padding) to distinguish between the text and the padded sequence to the BERT model. All the questions are padded to the maximum length of 64. This step is repeated for the test data questions.

Label encoding is performed to assign a unique integer for each category label. Boolean and resource category names remain the same. We flatten the category for `literal` by combining with its types (i.e, integer, string and date) to produce new labels (i.e, literal-integer, literal-string, literal-date). The 5 category labels are defined as a new column in the data frame.

Training The RoBERTa sequence classification model is built using the BERT base architecture with 12-layers, 768-hidden, 12-heads, 125M parameters. The classification model is defined with the number of classes required for the classification task, in our case 5. The training dataset (tokenized questions and encoded labels) is split in the ratio of 80:20 for training and validation. The classifier model is fine tuned on the training data for 4 epochs, using a batch size of 32 and learning rate=2e-5, with Adam optimizer. Accuracy is the measure of validation for classification task and categorical cross entropy as loss function. For both, Wikidata and DBpedia, we achieved a training accuracy of 98% and validation accuracy of 97%.

4.2 Resource Type Prediction

In case the category prediction, assigns the `resource` category to a question, the next steps are processed to predict the concrete answer types from the respective ontology.

Pre-processing For the different classification processes for DBpedia and Wikidata, we utilize the same pre-processing pipeline except for the first step: For DBpedia, we remove the prefix from the class labels, then apply lowercasing, and resolve the camel case format. For instance, `dbo:TelevisionShow` is transformed to *television show*. The Wikidata type labels remain unprocessed.

To extend the training data, several augmentation strategies might be applied. For our approach, we utilized the Copy-Paste method three times. Thus, the sample size of DBpedia is increased to 102,612 records and to 72,336 records for Wikidata.

The questions, class labels are assigned to blocks. We remove duplicates inside the block, create an indexation between the index of the label inside vocabulary and their block index. We add class labels from the vocabulary that do not exist in the training data. Then, the labels are merged inside the sentence randomly.

The questions syntactic order is shuffled in three different orders and recombined. We utilize lists consisting of: question, length, block, relations.Training data Q is appended into these lists separately and then the question list is split

into Q1 and Q2. The subsequent data is appended randomly into split question lists. Also, duplicates are removed.

For the tokenization step, we utilized the FastAiBertTokenizer within the fastai library as described in [5].

Training We utilized pre-trained model `BERT_BASE_UNCASED` which is not case-sensitive and has a much lower number of layers compared to the `BERT_LARGE` model. The loss function algorithm is used as binary cross entropy with logistic losses which applies a sigmoid activation layer to the output of the binary cross entropy layer to be mapped to 0 or 1. We achieved higher accuracies when using binary cross entropy compared to multi-label entropy. The maximum sequence length was set to 256, with a batch size of 32, and the model is trained for 20 iterations.

In terms of validation, we evaluated three different validation methods: static 80/20 split, static 99/1 split, and k fold cross validation with 3 folds. The results achieved are shown in the next section.

5 Evaluation

In this section, we present the evaluation results for our approach at the SMART challenge edition of ISWC 2021. The following metrics have been used for the evaluation:

- Accuracy: for category prediction (the percentage of questions classified in the correct category).
- Lenient NDCG@k: for DBpedia type prediction, measures the distance between the predicted type and the most specific type of the answer.
- MRR: for Wikidata type prediction, the Reciprocal Rank (RR) is reciprocal of the rank at which the first relevant type was retrieved, averaged across all types retrieved gives the Mean Reciprocal Rank.

The best results achieved by our two-staged approach are shown in Tables 2 and 3. The category prediction achieved an accuracy of 0.991 for both datasets and is best among all participants. As stated in the previous section, we evaluated different combinations of validation approaches and our data augmentation strategy. For DBpedia, we achieved the best results using the static 80/20 split for validation and the data augmentation strategy as described in Section 4.2. For Wikidata, data augmentation did not achieve an improvement of the results, therefore we skipped it. But, the cross validation using three folds achieved the best results compared to the static 80/20 and 99/1 splits.

6 Conclusion

In this paper, we presented our solution for an answer type prediction task as proposed by the SMART challenge co-located with ISWC 2021. We used two

Table 2: Results of our approach and the best competitor at SMART task 2021 challenge for DBpedia

	category accuracy	NDCG@5	NDCG@10
our approach	**0.991**	0.734	0.658
best competitor	0.984	**0.842**	**0.854**

Table 3: Results of our approach and the best competitor at SMART task 2021 challenge for Wikidata

	category accuracy	MRR
our approach	**0.991**	0.45
best competitor	0.98	**0.7**

different multi-label classification approaches for the category prediction and the type prediction for resource questions. Our results show that there is a significant difference between the necessary handling of the the datasets for the DBpedia and Wikidata. Obviously, Wikidata is a lot harder to train and achieve good results. As there is potential for improvements regarding our approach, we consider more intelligent data augmentation processes. The datasets are both very imbalanced regarding the distribution of type assignments especially for specific ontology classes. Therefore, it might be helpful to support the training process with more balanced data. Obviously, the difficult part of the challenge are resource questions and the prediction of the ontology classes. Future work will therefore include an investigation if the prediction of classes can be transformed independently from the requested ontology. Questions with similar type lists from both datasets could be consolidated – which would require a class mapping between the ontologies – and then the training process could benefit from a enhanced amount of different variants of resource questions with similar answer types.

7 Acknowledgements

This work was partially funded by the German Research Foundation (DFG) under grant no. SA 782/30-1 and STE 3033/1-1.

References

1. Barlas, I., Ginart, A., Dorrity, J.L.: Self-evolution in knowledge bases. In: IEEE Autotestcon, 2005. pp. 325–331. IEEE (2005)
2. Buzaaba, H., Amagasa, T.: Question answering over knowledge base: a scheme for integrating subject and the identified relation to answer simple questions. SN Computer Science **2**(1), 1–13 (2021)
3. Denny Vrandečić, M.K.: Wikidata: a free collaborative knowledgebase (2014)

4. Fu, B., Qiu, Y., Tang, C., Li, Y., Yu, H., Sun, J.: A survey on complex question answering over knowledge base: Recent advances and challenges (2020)
5. Howard, J., Gugger, S.: fastai: A layered API for deep learning. CoRR **abs/2002.04688** (2020), https://arxiv.org/abs/2002.04688
6. Jang, H., Oh, Y., Jin, S., Jung, H., Kong, H., Lee, D., Jeon, D., Kim, W.: Kbqa: Constructing structured query graph from keyword query for semantic search. In: Proceedings of the International Conference on Electronic Commerce. ICEC '17, Association for Computing Machinery, New York, NY, USA (2017). https://doi.org/10.1145/3154943.3154955, https://doi.org/10.1145/3154943.3154955
7. Kaufmann, E., Bernstein, A., Fischer, L.: Nlp-reduce: A naive but domainindependent natural language interface for querying ontologies. In: 4th European Semantic Web Conference ESWC. pp. 1–2. Springer Berlin (2007)
8. Lehmann, J.e.a.: Dbpedia – a large-scale, multilingual knowledge base extracted from wikipedia (2015)
9. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized bert pretraining approach (2019)
10. Luo, K., Lin, F., Luo, X., Zhu, K.: Knowledge base question answering via encoding of complex query graphs. In: Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing. pp. 2185–2194 (2018)
11. Malik, N., Sharan, A., Biswas, P.: Domain knowledge enriched framework for restricted domain question answering system. In: 2013 IEEE International Conference on Computational Intelligence and Computing Research. pp. 1–7 (2013). https://doi.org/10.1109/ICCIC.2013.6724163
12. Mihindukulasooriya, N., Dubey, M., Gliozzo, A., Lehmann, J., Ngomo, A.N., Usbeck, R.: Semantic answer type prediction task (SMART) at ISWC 2020 semantic web challenge. CoRR **abs/2012.00555** (2020), https://arxiv.org/abs/2012.00555
13. Mihindukulasooriya, N., Dubey, M., Gliozzo, A., Lehmann, J., Ngonga Ngomo, A.C., Usbeck, R., Rossiello, G., Kumar, U.: Semantic answer type and relation prediction task (smart 2021). arXiv (2022)
14. Pundge, A.M., Khillare, S., Mahender, C.N.: Question answering system, approaches and techniques: A review. International Journal of Computer Applications **141**(3), 0975–8887 (2016)
15. Qu, Y., Liu, J., Kang, L., Shi, Q., Ye, D.: Question answering over freebase via attentive rnn with similarity matrix based cnn. arXiv preprint arXiv:1804.03317 **38** (2018)
16. Setty, V., Balog, K.: Semantic answer type prediction using bert: Iai at the iswc smart task 2020 (2021)
17. Turing, A.M.: Computing machinery and intelligence. In: Parsing the turing test, pp. 23–65. Springer (2009)