Semantic Answer Type Prediction using BERT[*]
IAI at the ISWC SMART Task 2020

Vinay Setty and Krisztian Balog
{vinay.j.setty,krisztian.balog}@uis.no

University of Stavanger, Norway

Abstract. This paper summarizes our participation in the SMART Task of the ISWC 2020 Challenge. A particular question we are interested in answering is how well neural methods, and specifically transformer models, such as BERT, perform on the answer type prediction task compared to traditional approaches. Our main finding is that coarse-grained answer types can be identified effectively with standard text classification methods, with over 95% accuracy, and BERT can bring only marginal improvements. For fine-grained type detection, on the other hand, BERT clearly outperforms previous retrieval-based approaches.

Keywords: Answer type prediction · answer category classification · natural language understanding · question answering.

1 Introduction

The importance of being able to identify the types or semantic categories of answers requested has been long recognized in question answering (QA) research as a key step towards interpreting the meaning of natural language questions [4, 8]. This task may be performed either against a set of coarse-grained types (e.g., at the TREC QA track [9]) or against fine-grained type systems of knowledge bases, such as DBpedia [1, 5]. The Semantic Answer type prediction (SMART) task [7][1], organized as a challenge at the 2020 International Semantic Web Conference (ISWC '20) provides a large-scale evaluation platform for assessing answer type prediction both at course-grained and fine-grained taxonomical levels.

Specifically, given a natural language question as input, first a high-level answer type category is to be predicted, which can be one of `resource`, `literal`, or `boolean`. If the predicted category is `resource`, a more specific ontological class is to be provided, using the type system of DBpedia or Wikidata. If the predicted category is `literal`, it also has to be further classified as `number`, `date`, or `string`. In this paper, we refer to the task of coarse-grained answer detection as *category classification* and to the problem of fine-grained prediction

---

[*] Copyright © 2020 for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0). SeMantic AnsweR Type prediction task (SMART) at ISWC 2020 Semantic Web Challenge

[1] https://smart-task.github.io

Table 1. Example questions with respective answer categories and types (resources are based on DBpedia types).

Question	Category	Type
Who are the gymnasts coached by Amanda Reddin?	`resource`	`["dbo:Gymnast",` `"dbo:Athlete",` `"dbo:Person",` `"dbo:Agent"]`
How many superpowers does wonder woman have?	`literal`	`["number"]`
When did Margaret Mead marry Gregory Bateson?	`literal`	`["date"]`
Is Azerbaijan a member of European Go Federation?	`boolean`	`["boolean"]`

of (resource) types as *type prediction*. Table 1 shows some examples. As seen from the examples, answers for the resource category are provided as a ranked list of types.

The main research objectives in this work are to assess (1) How do neural approaches perform compared to traditional feature-based classification approaches on the category classification task? (2) How do neural classification approaches fare against well-establised (fusion-based) IR approaches on the type prediction problem? We find that (1) is essentially a "solved" problem. Our baseline SVM classifier with word unigrams as features achieved 95% accuracy. Neural approaches yield only minor improvements. As for (2), type prediction has previously been approached as a ranking problem, due to the large number of possible types ($\sim$760 types in DBpedia and $\sim$50k types in Wikidata) that rendered classification-based approaches infeasible. We draw on recent work on extreme multi-class classification and demonstrate substantial gains over the IR baselines. It appears that fine-grained type detection on Wikidata is more challenging than on DBpedia. However, the two are not directly comparable due to the different evaluation measures that are employed, which calls for further analysis.

Code and resources developed in this work are made publicly available at `https://github.com/iai-group/smart-task`.

2   Approach

We follow a two-phase approach. In the first phase, we perform *category classification*, that is, a supervised classifier predicts the high-level category of the answer type. Then, in the second phase, we perform *type prediction* to identify the top-k types for the questions for which answer type was predicted to be a resource. For category classification we use two classifiers: SVM with word unigrams as features and fine-tuned BERT (Section 2.1). Type prediction has previously been approached as a ranking task [1, 5], due to the large number of possible types. As an alternative, we can cast it as an extreme classification problem (Section 2.2).

2.1   Category Classification

We flatten the high-level categories into following five categories: `boolean`, `literal-number`, `literal-string`, `literal-date`, and `resource`. Since the category classification task is same for both DBpedia and Wikidata, we combine the training datasets for the two and predict the categories for their respective test datasets using the combined model.

Feature-based classification As a first approach to category classification, we use TF-IDF-weighted word unigrams as features. The vocabulary construction and IDF computations are based only on the training portion of the dataset, to avoid any assumptions on the test data. Our implementation is based on the `CountVectorizer` and `TFIDFVectorizer` classes from the `sklearn` library[2] with default parameters. We then train an SVM classifier with a linear kernel. We also experimented with using a Naive Bayes classifier, but decided to exclude that after observing inferior performance.

Neural approach As a second approach, we fine-tune a pre-trained BERT model (RoBERTa) [6] for a sequence classification task to classify the category. Our implementation uses the HuggingFace API[3] for fine-tuning and category classification.

2.2   Type Prediction

IR-based methods We employ two ranking-based approaches from [1], which were introduced for the task of identifying target types of (entity-bearing) search queries. These approaches are representatives of two main families of object ranking strategies, which have been termed as early and late fusion design patterns in [11]. According to the *type-centric* (TC, a.k.a. early fusion) approach, first a textual representation is built for each type by concatenating the descriptions of entities that are assigned that type. Then, these type description (pseudo) document can be ranked using standard IR models. Specifically, we use the DBpedia short abstracts of entities and then rank type documents using BM25. The second strategy is termed *entity-centric* (EC, a.k.a. late fusion). There, the top-k most relevant entities from the underlying knowledge base are retrieved using the question as a query. Then, the relevance score of a given type is computed by aggregating the relevance scores of entities with that type. We use BM25 as the underlying retrieval model and a "catch-all" entity representation, following the settings in [5]. The cut-off parameter k is chosen empirically based on the training set ($k = 20$).

---

[2] https://scikit-learn.org/
[3] https://huggingface.co/

Table 2. Dataset statistics.

Dataset	Questions		Categories		
	Train	Test	Boolean	Literal	Resource
DBpedia	17,571	4,393	2,799	5,188	9,584
Wikidata	18,251	4,571	2,139	4,429	11,683

Neural method Due to the large number of possible labels, using standard Transformer models is not feasible. Instead, we cast the type prediction task as an extreme multi-label text classification (XMC) problem: given a question as input text, return the top-k most relevant types from a large collection of possible types. Vanilla transformer models such as BERT [3], RoBERTa [6], and XLNet [10] are ineffective in this scenario due to the memory and computation requirements imposed by the large number of possible labels. This was also confirmed from our experiments that fine-tuning the above mentioned transformer models using the Huggingface framework exhausted all the memory on a 32GB Nvidia Tesla V100 GPU. While this may work on a GPU with larger memory, since we do not have access such a GPU we could not verify and it may still be computationally very expensive to train them. In addition to the computational limitations, as we show in Section 4, the types are very sparse with most of them having only a few training instances. In order to address these challenges, a model designed for XMC is essential. We use the recent solution to extend the transformer models for XMC coined *X-Transformers* [2] for this purpose, which shall be referred to as *XBERT* in the rest of the paper.

XBERT consists of three components:

1. Semantic Label Indexing (SLI), which performs hierarchical clustering on the labels to reduce the label space.
2. Deep Neural Matching (DNM), to fine-tune the Transformer models for each of the label clusters identified by SLI.
3. Ensemble Ranking (ER), which ranks the instances within the label clusters by training a linear ranker conditionally on the label clusters and the DNM Transformer's output.

3   Experimental Evaluation

In this section, we discuss our experimental setup, introduce the evaluation measures, and present our results.

3.1   Data

Table 2 presents descriptive statistics for the two datasets, DBpedia and Wikidata. We notice that Wikidata has slightly more `resource` than `literal` answer types, compared to DBpedia.

3.2 Methods

The following methods are compared:

- SVM: Support Vector Machine for category classification
- BERT: RoBERTa for category classification
- XBERT: X-Transformers for type prediction
- IR/TC: Type-centric IR approach for type prediction
- IR/EC: Entity-centric IR approach for type prediction

We train all neural models on a single Nvidia Tesla V100 GPU with 32GB memory.

3.3 Evaluation Metrics

Category classification is evaluated in terms of classification accuracy. *Type prediction* is cast as a ranking task and is evaluated using rank-based metrics. It, however, considers only those questions that fall into the `literal` or `resource` answer categories. Furthermore, evaluation is performed differently for DBpedia and for Wikidata, given the nature of their respective type taxonomies. Types in the DBpedia Ontology are organized hierarchically, up to 7 levels deep. There, a graded evaluation metric, Normalized Discounted Cumulative Gain (NDCG@k), is used. Specifically:

- For `literal` answer types, only a single predicted type is considered that can be either correct (NDCG=1) or incorrect (NDCG=0).
- For `resource` answer types, a ranked list of top-k ontology classes is considered and evaluated in terms of lenient NDCG@k with linear decay [1]. The gain for a given predicted type is 0 if it is not on the same path with any of the gold types, and otherwise it is $1 - d(t, t_q)/h$, where $d(t, t_q)$ is the distance between the predicted type and the closest matching gold type in the type hierarchy, with h being the maximum depth of the type hierarchy.

In case of Wikidata, the type hierarchy is rather flat. Therefore, type prediction is evaluated using a binary notion of relevance, with Mean Reciprocal Rank (MRR) as the metric.

We report results on the training dataset, using 5-fold cross-validation. For our official submissions, we also report the performance on the test set.

3.4 Results

Category Classification It can be seen from the results in Table 3 that both feature-based and neural approaches perform quite well for category classification. BERT has a slight advantage over SVM. We hypothesize that due to the clear patterns which the models can learn, the high-level category classification is a fairly easy task and hence the high accuracy scores. However, most mistakes occur for the `resource` class, which is the majority class in both datasets.

Table 3. Category classification results, measured in terms of Accuracy. Best scores for each dataset are in boldface.

Dataset	Method	Train	Test
DBpedia	SVM	0.958	0.964
	BERT	**0.970**	**0.977**
Wikidata	SVM	0.956	0.960
	BERT	**0.964**	**0.970**

Table 4. Type prediction results on DBpedia. Best scores are in boldface.

Method	Train		Test	
	NDCG@5	NDCG@10	NDCG@5	NDCG@10
BERT-IR/TC	0.492	0.509	-	-
BERT-IR/EC	0.483	0.503	-	-
SVM-XBERT	0.773	0.744	0.790	0.778
BERT-XBERT	**0.776**	**0.747**	**0.804**	**0.793**

Table 5. Type prediction results on Wikidata (measured in terms of MRR). Best scores are in boldface.

Method	Train data	Test data
SVM-XBERT	0.66	0.67
BERT-XBERT	**0.67**	**0.68**

Type Prediction Since different metrics are used for DBpedia and Wikidata, we report results on the two datasets separately, in Tables 4 and 5, respectively. Recall, that (stage-two) type prediction is applied on top of (stage-one) category classification (SVM or BERT) and is only carried out when the predicted category is `resource`. We thus prefix the method names in the result tables with SVM- or BERT- to indicate how category classification was performed.

On DBpedia (Table 4), XBERT clearly outperforms the IR approaches. We attribute this to the fact that XBERT is tailored for XMC problem which can deal with large number of types and sparsity with tail resource types. The slight difference between SVM-XBERT and BERT-XBERT is due to the mistakes made by SVM in category classification. Given the large advantage of XBERT over the IR approaches, our official submissions on Wikidata (Table 5) only considered the former. It should nevertheless be noted that the IR approaches are unsupervised methods that do not need any training data. Supervised alternatives have shown to perform significantly better [5]. We leave that comparison to future work.

Table 6. Top-10 mistakes in type prediction by the BERT-XBERT model for DBpedia and Wikidata.

DBpedia Type	#Total	#Errors	Wikidata Type	#Total	#Errors
dbo:Person	2713	184	natural person	1901	214
dbo:Country	751	86	omnivore	1901	214
dbo:State	490	59	person	1963	204
dbo:Organisation	1401	45	political territorial entity	570	120
dbo:Media	188	39	country	685	119
dbo:Company	440	38	state	692	119
dbo:Activity	204	38	organization	426	110
dbo:Work	898	26	class	375	103
dbo:Band	98	25	community	326	86
dbo:Profession	97	25	big city	235	82

4 Error Analysis

In this section, we analyze the errors made by the our best performing approach, BERT-XBERT. First, we look at resource types where most errors occur. That is, types which are present in the gold labels but are missing from the predicted labels. Table 6 shows the top-10 errors in type prediction for DBpedia and Wikidata, together with their total instance counts. Ideally, we would expect that the number of mistakes to be directly proportional to the total frequency of the resource type. In DBpedia, some types such as `dbo:State`, `dbo:Activity`, `dbo:Band`, and `dbo:Profession` break this pattern. Similarly in Wikidata, `natural person`, `political territorial entity`, and `big city` are some of types with which the BERT-XBERT model struggles.

In Table 7, we show anecdotal examples of the mistakes made by the BERT-XBERT approach. Most of these errors are due to irrelevant types returned in the result list. In several cases, the predicted labels do contain the the gold label but place them at lower ranks, which affects the NDCG and MRR scores. In some cases the predicted labels are appropriate, even though they do not exactly match the gold labels. For example, for the last question in Table 7, `publication` is one of the gold labels, which is not predicted, but `written work` and `periodical` are still relevant among the predicted labels. We also spotted several instances with double questions such as "What conflict occurred in Philoctetes and who was involved?" and questions with grammatical errors and typos.

5 Conclusions

In this paper, we presented our solution for the SMART Task challenge of ISWC 2020, which was the best performing approach on both datasets and tasks, across all evaluation metrics. Our findings suggest that for coarse-grained category prediction, simple feature-based approaches are quite effective with over

Table 7. Example questions from DBpedia (top two) and Wikidata (bottom two) with respective gold and predicted type labels (by BERT-XBERT method).

Question	Gold types	Predicted types
What is the title of Kakae ?	`["dbo:Settlement",` `"dbo:Island"]`	`["dbo:Agent", "dbo:Media",` `"dbo:Organisation",` `"dbo:PersonFunction",` `"dbo:Profession",` `"dbo:Person", "dbo:Island",` `"dbo:University",` `"dbo:Bone", "dbo:Factory"]`
List the destination of No-vair International Airways ?	`["dbo:Sea",` `"dbo:Country",` `"dbo:River",` `"dbo:Continent"]`	`["dbo:Country",` `"dbo:Location", "dbo:Place",` `"dbo:PopulatedPlace",` `"dbo:Continent",` `"dbo:Airport",` `"dbo:MeanOfTransportation",` `"dbo:Aircraft",` `"dbo:Infrastructure",` `"dbo:ArchitecturalStructure"]`
which cola starts with the letter p	`["soft drink",` `"trademark",` `"carbonated` `beverage",` `"non-alcoholic` `beverage",` `"symbol", "class",` `"protected name"]`	`["non-alcoholic beverage",` `"carbonated beverage", "soft` `drink", "trademark", "food",` `"long gun", "goods", "dish",` `"cyclic process", "tea"]`
What periodical literature does Delta Air Lines use as a moutpiece?	`["publication",` `"recurring",` `"intellectual` `work", "text",` `"communication` `medium", "serial"]`	`["organization", "creative` `work", "written work",` `"text", "magazine", "media` `enterprise", "newspaper",` `"communication medium",` `"genre", "periodical"]`

95% accuracy, while sophisticated neural Transformer architectures only improve marginally. For fine-grained type prediction, on the other hand, Transformer models for extreme multilabel classification clearly outperform retrieval-based approaches.

Our future work concerns an in-depth analysis of the results on DBpedia vs. Wikidata, to understand the differences and modeling requirements for small and hierarchical (DBpedia) vs. large and shallow (Wikidata) type taxonomies.

Bibliography

[1] K. Balog and R. Neumayer. Hierarchical target type identification for entity-oriented queries. In *Proceedings of the 21st ACM international conference on Information and knowledge management*, CIKM '12, pages 2391–2394, 2012.

[2] W.-C. Chang, H.-F. Yu, K. Zhong, Y. Yang, and I. S. Dhillon. Taming pretrained transformers for extreme multi-label text classification. In *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pages 3163–3171, 2020.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pages 4171–4186, 2019.

[4] D. A. Ferrucci, E. W. Brown, J. Chu-Carroll, J. Fan, D. Gondek, A. Kalyanpur, A. Lally, J. W. Murdock, E. Nyberg, J. M. Prager, N. Schlaefer, and C. A. Welty. Building Watson: An overview of the DeepQA project. *AI Magazine*, 31(3):59–79, 2010.

[5] D. Garigliotti, F. Hasibi, and K. Balog. Target type identification for entity-bearing queries. In *Proceedings of the 40th International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '17, pages 845–848, 2017.

[6] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov. Roberta: A robustly optimized bert pre-training approach. *arXiv preprint arXiv:1907.11692*, 2019.

[7] N. Mihindukulasooriya, M. Dubey, A. Gliozzo, J. Lehmann, A.-C. N. Ngomo, and R. Usbeck. SeMantic AnsweR Type prediction task (SMART) at ISWC 2020 Semantic Web Challenge. *CoRR/arXiv*, abs/2012.00555, 2020. URL `https://arxiv.org/abs/2012.00555`.

[8] T. Shen, X. Geng, Q. Tao, D. Guo, D. Tang, N. Duan, G. Long, and D. Jiang. Multi-task learning for conversational question answering over a large-scale knowledge base. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 2442–2451, 2019.

[9] E. M. Voorhees. The trec question answering track. *Nat. Lang. Eng.*, 7(4): 361–378, Dec. 2001.

[10] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le. Xlnet: Generalized autoregressive pretraining for language understanding. In *Advances in neural information processing systems*, pages 5753–5763, 2019.

[11] S. Zhang and K. Balog. Design patterns for fusion-based object retrieval. In *Proceedings of the 39th European conference on Advances in Information Retrieval*, ECIR '17, pages 684–690. Springer, 2017.