The Role of Output Vocabulary in T2T LMs for SPARQL Semantic Parsing

Debayan Banerjee [†1], Pranav Ajit Nair [†2], Ricardo Usbeck[1], and Chris Biemann[1]

[1]Universität Hamburg, Hamburg, Germany
[1]{firstname.lastname}@uni-hamburg.de
[2]Indian Institute of Technology (BHU), Varanasi, India
[2]pranavajitnair.cse18@itbhu.ac.in

Abstract

In this work, we analyse the role of output vocabulary for text-to-text (T2T) models on the task of SPARQL semantic parsing. We perform experiments within the the context of knowledge graph question answering (KGQA), where the task is to convert questions in natural language to the SPARQL query language. We observe that the query vocabulary is distinct from human vocabulary. Language Models (LMs) are pre-dominantly trained for human language tasks, and hence, if the query vocabulary is replaced with a vocabulary more attuned to the LM tokenizer, the performance of models may improve. We carry out carefully selected vocabulary substitutions on the queries and find absolute gains in the range of 17% on the GrailQA dataset.

1 Introduction

Knowledge Graph Question Answering (KGQA) is the task of finding answers to questions posed in natural language, using triples present in a KG. Typically the following steps are followed in KGQA: 1) Objects of interest in the natural language question are detected and linked to the KG in a step called entity linking. 2) The relation between the objects is discovered and linked to the KG in a step called relation linking. 3) A formal query, usually SPARQL[1], is formed with the linked entities and relations. The query is executed on the KG to fetch the answer.

Our focus in this work is the query building phase, henceforth referred to as KGQA semantic parsing. The motivation of our work stems from Banerjee et al. (2022), where minor vocabulary substitutions to handle non-printable special characters for T5 (Raffel et al., 2020) produced better results on the task of SPARQL semantic parsing. In this work, we extend the idea and replace the entire SPARQL vocabulary with alternate vocabularies.

As in Banerjee et al. (2022), we replace certain special characters in the SPARQL vocabulary, such as { , } with textual identifiers, as T5 is known to have problems dealing with these special characters (Banerjee et al., 2022). We call this a masked query, and in this work, we test the ability of the models to generate this masked query, given the natural language question as input.

A sample question, the original SPARQL query, and the corresponding masked query are as shown below (for the Wikidata KG (Vrandečić and Krötzsch, 2014)) :

*Is it true that an Olympic-size swimming pool's operating temperature is equal to **22.4** ?*

```
ASK WHERE
{
  wd:Q2084454 wdt:P5066 ?obj
  filter(?obj = 22.4)
}
```

```
ASK WHERE
OB
  ent0 rel0 ?obj
  filter ( ?obj = 22.4 )
CB
```

In the era of pre-trained Language Models (LMs) (Devlin et al., 2019; Raffel et al., 2020) it is common practice to fine-tune models on custom downstream datasets. This requires supervised training which results in modification of weights of the models using some training algorithm. More recently, the technique of prompting of language models (Brown et al., 2020; Shin et al., 2020) has been developed, which elicits the desired response from a LM through a task description and a few input-output examples. Brown et al. (2020) shows that such a strategy works better for larger models. It has however been observed that prompt design is brittle in behaviour and displays sensitivity to the

[†]The authors contributed equally to this work
[1]https://www.w3.org/TR/rdf-sparql-query/

Findings of the Association for Computational Linguistics: ACL 2023, pages 12219–12228
July 9-14, 2023

exact phrase (Shin et al., 2020).

A more recent innovation is that of prompt tuning (Lester et al., 2021), where the task-specific prompt is learnt on a smaller external neural network. The gradients are computed and flow through the LM, but leave the weights of the LM itself unchanged. Instead, the weights of the prompt tuning network change and produce a custom and continuous prompt which produces the desirable response from the LM.

A similar method is prefix tuning (Li and Liang, 2021), which is known to perform better for generation tasks (Ma et al., 2022). In this method, the original inputs and outputs are kept the same, but the input is pre-pended with a continuous prefix learnt in the external network. This prefix allows the model to understand the exact task to be performed by it.

As primary contribution, in this work, we perform an analysis of how the complexity of output vocabularies affects the performance on the KGQA semantic parsing task for prefix and fine-tuned language models. Code and data can be found at https://github.com/debayan/sparql-vocab-substitution.

2 Related Work

A study of low-resource semantic parsing using prompt tuning was performed by Schucher et al. (2022) on the Top v2 (Chen et al., 2020) and Overnight (Wang et al., 2015) datasets. Prompt tuning, while not the same as prefix tuning, still keeps the LM weights frozen while the prompts are learnt on an external network. In their experiments, they perform a single kind of vocabulary substitution but find no noticeable performance improvements. No specific study is made of the change in performance with vocabularies of varying complexities, which is a task we undertake. Another difference is that we perform experiments in the high-resource use case as opposed to low-resource.

Another work which is similar to ours is Sun et al. (2022), where the authors experiment with prefix tuning on the task of semantic parsing, and find problems with non-standard vocabularies of logical forms. In their case, they work with the TOP v2 (Chen et al., 2020) and PIZZA (Arkoudas et al., 2022) datasets. The keywords in those datasets consist of words joined by underscores (eg: IN:GET_REMINDER_DATA_TIME), which poses a problem for the sub-word tokenizer of the transformer based models. They find that fine tuning a model on these datasets outperforms prefix-tuning by a large margin. However, when they add the non-standard keywords to the tokenizer vocabulary and re-train the tokenizer to generate new embeddings for these keywords, fine tuning and prefix tuning perform at par. Our work is different in a few respects: firstly, due to the specific research focus of our group, we experiment with a semantic parsing dataset for KGQA, namely GrailQA (Gu et al., 2021). Secondly, instead of retraining the tokenizer, we perform a simpler procedure of pre-processing the dataset by replacing the current vocabulary with a new vocabulary. We then train the models on this modified dataset, and as a post-processing step, substitute back the original vocabulary in place of the new vocabulary.

3 Prefix Tuning

Prefix tuning prepends a set of tunable weights to every key-value pair in the transformer attention. The transformer attention is represented as follows:

$$\text{attn}(Q, K, V) = \text{softmax}(\frac{Q \cdot K^\top}{\sqrt{d}})V \quad (1)$$

where the query Q, key K and value V are obtained through affine transformations on the input. d represents the model dimension. Prefix tuning modifies the transformer attention by adding tunable prefixes to K and V, thereby modifying K as $K' = [h_K; K]$ and V as $V' = [h_V; V]$. Here h_K and h_V represent the key prefix and the value prefix respectively.

Following Li and Liang (2021) we model these prefixes using a two layer MLP as follows:

$$\begin{aligned} h_K &= W_{K,2} f(W_{K,1} E + b_{K,1}) + b_{K,2} \\ h_V &= W_{V,2} f(W_{V,1} E + b_{V,1}) + b_{V,2} \end{aligned} \quad (2)$$

where $W \in \mathbb{R}^{d \times d}$ and $b \in \mathbb{R}^d$ are trainable weights and biases respectively. $E \in \mathbb{R}^{C \times d}$ is a trainable embedding matrix with C as the prefix length.

4 Models and Experimental Setup

We carry out prefix-tuning and fine-tuning experiments with two versions of the T5 model: namely T5-Small (60 million parameters) and T5-Base (220 million parameters). Questions are fed as input during training while masked SPARQL queries, as described in Section 1, are provided as labels for supervision.

	GrailQA					
	T5-Small		T5-Base			
	PT	FT	PT	FT	TSVS	ALFL
char8	74.03	86.57	82.65	86.72	306	263
char4	76.43	87.09	84.92	87.10	159	141
char2	83.29	91.49	89.83	92.30	90	87
char1	**84.89**	**92.13**	**91.24**	**92.61**	57	57
dictionary	82.57	91.95	90.93	92.48	49	44
original	67.10	74.08	73.06	74.45	124	125

Table 1: Exact match percentages for generated masked SPARQL queries. Best performance is always found in substituted vocabularies. For **char** settings, accuracy drops as vocabulary and query lengths increase. TSVS = Tokenizer specific vocabulary size, ALFL = Average logical form length, PT = Prefix Tuning, FT = Fine Tuning

For evaluation, we use the exact-match metric. A generated query is matched token by token, while ignoring white-spaces, to the gold query. The percentage of queries matched is reported.

4.1 Hyper-parameters and Implementation Details

Throughout our experiments, the prefix length is fixed to 50. For prefix tuning experiments we use the Adafactor (Shazeer and Stern, 2018) optimizer with a constant learning rate of 0.001. Fine-tuning experiments are optimized through AdamW (Loshchilov and Hutter, 2019) with a square root decay schedule, a maximum learning rate of 0.0015 and a linear warm-up of 5000 steps. Our code is implemented with HuggingFace Transformers[2] (Wolf et al., 2020) and OpenPrompt[3] (Ding et al., 2022). T5-Small experiments were run on 12GB Nvidia GTX-1080 and RTX-2080 GPUs, and T5-Base experiments were run on 48GB Nvidia RTX-A6000. For fine-tuning, we run each training thrice with three separate seeds for 120 epochs each. For prompt tuning we do the same for 400 epochs. We report the inference results of these trained models on the test sets of the respective datasets.

5 Vocabulary

The original vocabulary of the GrailQA dataset consists of 48 words. The T5 tokenizer splits these words into 124 sub-words. This tokenizer specific vocabulary size (TSVS) is seen in the last column of Table 1. In the next column, the original average logical form (SPARQL query) length can be seen as 125 tokenized sub-words.

We wish to see how a new output vocabulary affects performance, and as a result, we construct a set of special vocabularies and substitute them in-place of the original SPARQL vocabulary. With reference to the settings in Table 1, each vocabulary is as described below:

original The masked SPARQL queries remain as they are. No replacement of the original SPARQL keywords is made with an alternate vocabulary.

dictionary The SPARQL keywords are replaced with a vocabulary of English words. For example, SELECT may be replaced with DOG, [may be replaced with CAT etc. During the pre-training phase a LM is likely to have seen such words far more frequently than the SPARQL keywords. This mode tests how the model behaves when the output vocabulary is comprised of well known English words.

char1 The SPARQL keywords are replaced with a single character of the English alphabet, for example, SELECT is replaced with A, WHERE is replaced with B. Additionally, numerical digits from 1-9 are used, and if the size of vocabulary demands more, we add single length special characters, such as * and $.

char2, **char4** and **char8** settings apply vocabulary substitution of 2, 4 and 8 character lengths chosen randomly, constituted from the characters A-Z and digits 0-9. For example, a typical **char8** substitution would be SELECT replaced by ATYZGFSD. This setting is designed to test the behaviour of the models when asked to produce more number of tokens per original-vocabulary word. A sample of a question, the SPARQL and the corresponding substitutions is provided in the Appendix in Table 2.

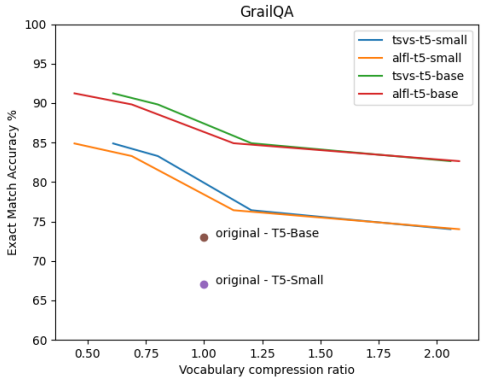
[2] https://github.com/huggingface/transformers

[3] https://github.com/thunlp/OpenPrompt

6 Datasets

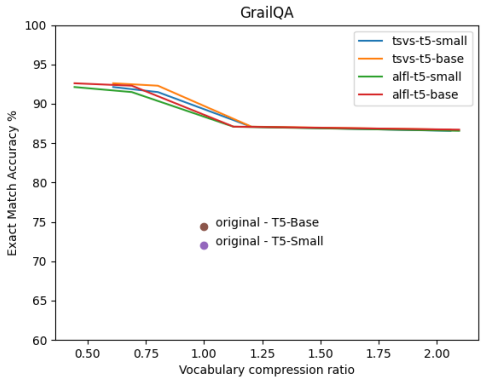
For our experiments, we require a dataset which contains a mapping of natural language questions to their corresponding logical forms and is large in size, since we test the high resource use-case.

GrailQA [4] is based on the Freebase knowledge graph (Bollacker et al., 2008) and consists of 64,331 questions designed to test three levels of generalisation, ie, i.i.d, compositional and zero-shot. For our purposes, we split the train set itself to three parts, since we are not interested in testing compositional generalisation aspects of the test set of this dataset. We are left with the following configuration: test: 8868, dev: 4434, train: 31035.



(a)

Figure 1: Prefix tuning accuracy drops as vocabulary and query lengths increase for **char** settings. TSVS = Tokenizer specific vocabulary size, ALFL = Average logical form length



(a)

Figure 2: Fine-tuning accuracy drop is more gradual when compared to prefix tuning, and the performance of T5-Small and T5-Base are similar. TSVS = Tokenizer specific vocabulary size, ALFL = Average logical form length

[4] https://dki-lab.github.io/GrailQA/

7 Analysis

As seen in Table 1, the best performance for prefix and fine tuning is achieved for substituted vocabularies. The original vocabulary lags behind in general, which points to the finding, that the choice of an appropriate vocabulary improves performance for semantic parsing. Further, among the substituted vocabularies, the setting **char8** performs the worst, which signifies the adverse role of the extra decoding load of this vocabulary on the performance of the model.

This finding is different from that of Schucher et al. (2022), who find their `in-vocab` setting performing no better overall. They attribute it to the substitutions possibly masking the meanings of the intents, for their given dataset. On the contrary, we find significant gains for GrailQA. It must be noted however, that we perform high-resource prefix tuning while they perform low-resource prompt tuning, and hence results may differ.

As seen in Figure 1, for the **char** settings, as the size of vocabulary increases, the prefix tuning accuracy drops. In the said figure, we define vocabulary compression ratio as the size of the new vocabulary divided by the size of the original vocabulary. Apart from vocabulary size, the query length also matters. We dual-define vocabulary compression ratio as the size of query length after substitution of new vocabulary divided by size of original query length, and plot on the same graph.

When compared to the fine-tuning plot (Figure 2), prefix tuning has a steeper drop in accuracy, and the performance for T5-Small and T5-Base vary more significantly. It leads to the finding that fine-tuning is less sensitive to vocabulary changes, and the difference in model sizes between T5-Small and T5-Base also seems to matter less.

In Figures 1 and 2, it can be seen that the **original** setting for the masked SPARQL vocabularies produce accuracies which are below the **char** family vocabulary curves. It suggests that vocabulary compression ratio alone is not a deciding factor in accuracy. If the vocabulary family changes from SPARQL to characters, there is an initial shift in accuracy, and after that the complexity of the character vocabulary further affects the accuracy.

In Table 1, the **dictionary** setting performs slightly worse than the **char1** setting, although it has lower TSVS and ALFL. This suggests that the vocabulary size and query length are not the only factors that affect the eventual accuracy. Perhaps

the frequency of the tokens seen by the model during the pre-training task plays a role. It is likely that the model has encountered, during pre-training, single characters a far larger number of times than the words used in **dictionary** vocabulary.

8 Error Analysis

We performed an error analysis on a sample of 100 randomly selected questions which produced an incorrect output. In the **original** setting, roughly 50% errors were due to the presence of non-printable characters in the query (eg: ^). We found that in the initial masked query, while we had replaced some non-printable characters in the pre-processing stage (eg: {, }), we had not managed to replace the full set of non-printable characters. The original T5 paper mentions curly braces as one of the class of tokens that are not present in the pre-training corpus, however, a comprehensive list of the tokens that do not work with T5, or work with limited efficiency, is not available. In this scenario, it seems that a better approach is to replace the entire vocabulary with one that is entirely known to T5, for example, English words. When comparing errors made by **original**, that were fixed by **dictionary** and **char1**, we observed that roughly 30% of the cases were of variable placement, where the variable placeholders like `ent0`, `rel0` were found to be in the wrong order in the output query in the **original** setting. Rest of the corrections belonged to the category of syntax errors. This points to the finding that alternate vocabularies improve the ability of T5 to correctly produce logical forms from a semantic perspective.

To analyse the effect of increasing complexity of vocabulary, we compare 100 randomly selected errors made by **char8** with **char2**. In both these settings, no character is non-printable, and the only errors are either syntax errors, variable placement errors, structural errors or intent errors. Out of the 100 questions, 90 were found to be correct in **char2** setting. In the remaining 90 in the **char8** setting, the highest proportion of errors belonged to syntax (where the query is malformed). The next most prominent class of errors belonged to variable placement, followed by structural errors (eg: two triples instead of three). The major takeaway from this analysis is that for **char2** there were no syntax errors, while in **char8** there are a significant number of such errors.

9 Conclusion

In this work we carried out experiments with new output vocabularies, where we carefully substituted the original members of the vocabulary with the new ones. We found that when the original SPARQL vocabulary is replaced with words from an alternate vocabulary closer to the T5 tokenizer vocabulary, the model consistently perform better.

As a contribution, we believe that our findings will enable researchers in the field of semantic parsing to deploy smaller models with a modified vocabulary and still find satisfactory performance. This would, in the longer term, lead to energy savings.

As future work, we would like to explore the behaviour of the same models in more depth using attention maps. Moreover, the significant shift in initial performance on changing vocabulary from **original** to **char** and **dictionary** demands further investigation. Similarly, the relatively lower performance of the **dictionary** setting when compared to **char1** setting, in spite of having lower tokenized vocabulary size (TSVS) needs to be investigated further. Perhaps sub-words which are seen more frequently during pre-training task of the LM perform better when substituted into the semantic parsing output vocabulary.

10 Limitations

We found that prefix tuning takes much longer to converge when compared to fine tuning, and for T5-Base, it takes around 10 days on a 48 GB GPU to complete tuning for a single setting in Table 1. Due to limitation of resources and with an aim to save energy, we did not conduct experiments with larger models such as T5-Large, T5-XL etc. We also did not perform experiments with smaller splits of the same datasets, which could have given further insights on how model performance varies when training data size is less.

References

Konstantine Arkoudas, Nicolas Guenon des Mesnards, Melanie Rubino, Sandesh Swamy, Saarthak Khanna, Weiqi Sun, and Khan Haidar. 2022. PIZZA: A new benchmark for complex end-to-end task-oriented parsing. *arXiv preprint arXiv:2212.00265*.

Debayan Banerjee, Pranav Ajit Nair, Jivat Neet Kaur, Ricardo Usbeck, and Chris Biemann. 2022. Modern Baselines for SPARQL Semantic Parsing. In *Proceedings of the 45th International ACM SIGIR Con-*

ference on Research and Development in Information Retrieval, SIGIR '22, page 2260–2265, New York, NY, USA. Association for Computing Machinery.

Kurt Bollacker, Colin Evans, Praveen Paritosh, Tim Sturge, and Jamie Taylor. 2008. Freebase: A collaboratively created Graph Database for structuring human knowledge. In *SIGMOD '08: Proceedings of the 2008 ACM SIGMOD international conference on Management of data*, pages 1247–1250, New York, NY, USA. ACM.

Tom B. Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, Sandhini Agarwal, Ariel Herbert-Voss, Gretchen Krueger, Tom Henighan, Rewon Child, Aditya Ramesh, Daniel M. Ziegler, Jeffrey Wu, Clemens Winter, Christopher Hesse, Mark Chen, Eric Sigler, Mateusz Litwin, Scott Gray, Benjamin Chess, Jack Clark, Christopher Berner, Sam McCandlish, Alec Radford, Ilya Sutskever, and Dario Amodei. 2020. Language models are few-shot learners. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS'20, Red Hook, NY, USA. Curran Associates Inc.

Xilun Chen, Asish Ghoshal, Yashar Mehdad, Luke Zettlemoyer, and Sonal Gupta. 2020. Low-Resource Domain Adaptation for Compositional Task-Oriented Semantic Parsing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 5090–5100, Online. Association for Computational Linguistics.

Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2019. 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, Minneapolis, Minnesota. Association for Computational Linguistics.

Ning Ding, Shengding Hu, Weilin Zhao, Yulin Chen, Zhiyuan Liu, Haitao Zheng, and Maosong Sun. 2022. OpenPrompt: An Open-source Framework for Prompt-learning. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics, ACL 2022 - System Demonstrations, Dublin, Ireland, May 22-27, 2022*, pages 105–113. Association for Computational Linguistics.

Yu Gu, Sue Kase, Michelle Vanni, Brian Sadler, Percy Liang, Xifeng Yan, and Yu Su. 2021. Beyond I.I.D.: Three Levels of Generalization for Question Answering on Knowledge Bases. In *Proceedings of the Web Conference 2021*, WWW '21, page 3477–3488, New York, NY, USA. Association for Computing Machinery.

Brian Lester, Rami Al-Rfou, and Noah Constant. 2021. The power of scale for parameter-efficient prompt tuning. In *Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing*, pages 3045–3059, Online and Punta Cana, Dominican Republic. Association for Computational Linguistics.

Xiang Lisa Li and Percy Liang. 2021. Prefix-Tuning: Optimizing Continuous Prompts for Generation. In *Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing (Volume 1: Long Papers)*, pages 4582–4597, Online. Association for Computational Linguistics.

Ilya Loshchilov and Frank Hutter. 2019. Decoupled Weight Decay Regularization. In *7th International Conference on Learning Representations, ICLR 2019, New Orleans, LA, USA*. OpenReview.net.

Fang Ma, Chen Zhang, Lei Ren, Jingang Wang, Qifan Wang, Wei Wu, Xiaojun Quan, and Dawei Song. 2022. XPrompt: Exploring the Extreme of Prompt Tuning. *arXiv preprint arXiv:2210.04457*.

Colin Raffel, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. 2020. Exploring the Limits of Transfer Learning with a Unified Text-to-Text Transformer. 21(1).

Nathan Schucher, Siva Reddy, and Harm de Vries. 2022. The power of prompt tuning for low-resource semantic parsing. In *Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 148–156, Dublin, Ireland. Association for Computational Linguistics.

Noam Shazeer and Mitchell Stern. 2018. Adafactor: Adaptive learning rates with sublinear memory cost. In *International Conference on Machine Learning*, pages 4596–4604. PMLR.

Taylor Shin, Yasaman Razeghi, Robert L. Logan IV, Eric Wallace, and Sameer Singh. 2020. AutoPrompt: Eliciting Knowledge from Language Models with Automatically Generated Prompts. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 4222–4235, Online. Association for Computational Linguistics.

Weiqi Sun, Haidar Khan, Nicolas Guenon des Mesnards, Melanie Rubino, and Konstantine Arkoudas. 2022. Unfreeze with Care: Space-Efficient Fine-Tuning of Semantic Parsing Models. In *Proceedings of the ACM Web Conference 2022*, WWW '22, page 999–1007, New York, NY, USA. Association for Computing Machinery.

Denny Vrandečić and Markus Krötzsch. 2014. Wikidata: A Free Collaborative Knowledgebase. volume 57, page 78–85, New York, NY, USA. Association for Computing Machinery.

Yushi Wang, Jonathan Berant, and Percy Liang. 2015. Building a Semantic Parser Overnight. In *Proceedings of the 53rd Annual Meeting of the Association*

for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 1: Long Papers), pages 1332–1342, Beijing, China. Association for Computational Linguistics.

Thomas Wolf, Lysandre Debut, Victor Sanh, Julien Chaumond, Clement Delangue, Anthony Moi, Pierric Cistac, Tim Rault, Rémi Louf, Morgan Funtowicz, Joe Davison, Sam Shleifer, Patrick von Platen, Clara Ma, Yacine Jernite, Julien Plu, Canwen Xu, Teven Le Scao, Sylvain Gugger, Mariama Drame, Quentin Lhoest, and Alexander M. Rush. 2020. Transformers: State-of-the-Art Natural Language Processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations*, pages 38–45, Online. Association for Computational Linguistics.

A Samples

	GrailQA
Question	Military airfield is the type for what airport ?
SPARQL	`SELECT DISTINCT ?x0 WHERE { ?x0 :type.object.type :aviation.airport . VALUES ?x1 { :m.0199qf } ?x0 :aviation.airport.airport_type ?x1 . FILTER ( ?x0 != ?x1 ) }`
Masked Query (**original** setting)	`SELECT DISTINCT ?x0 WHERE OB ?x0 :type.object.type rel0 . VALUES ?x1 OB ent0 CB ?x0 rel1 ?x1 . FILTER ( ?x0 != ?x1 ) CB`
dictionary	`banana compound boy nation rain boy catastrophe elementary flower teeth today rain jacket case boy fog today flower duck folk boy chart today concede case`
char1	`- 1 A Y $ A : O % L J $ G S A | J % 0 M A + J X S`
char2	`UY SJ 0X 6L VZ 0X 5G JO SE 5Z QB VZ QJ 8O 0X FT QB SE RU 2K 0X WY QB I5 8O`
char4	`53IY 3UQZ JKMQ CEK2 5DZV JKMQ KRDN 1G8E ZC5C 5ILL 3JBD 5DZV X5XB YMG5 JKMQ ZVGC 3JBD ZC5C 87O2 DE3Z JKMQ TU76 3JBD 049K YMG5`
char8	`WDEUTG57 L741BHJP ORWDXYPH 6L05N8AS ZLZXSARH ORWDXYPH K4GR9TPQ 797G3PGO V13Y1EFE PQMAIPQ4 MLN1V72G ZLZXSARH KPHC8I2N WG0XRTYG ORWDXYPH ZF82YUH8 MLN1V72G V13Y1EFE 41O2LA2M F1SANW03 ORWDXYPH 4R26K1BW MLN1V72G TD9BSKSN WG0XRTYG`

Table 2: An example of a question from GrailQA, with the corresponding SPARQL query, and how they look once new vocabularies are substituted.

ACL 2023 Responsible NLP Checklist

A For every submission:

☑ A1. Did you describe the limitations of your work?
9

☑ A2. Did you discuss any potential risks of your work?
9

☑ A3. Do the abstract and introduction summarize the paper's main claims?
1

☒ A4. Have you used AI writing assistants when working on this paper?
Left blank.

B ☒ Did you use or create scientific artifacts?

Left blank.

☐ B1. Did you cite the creators of artifacts you used?
Not applicable. Left blank.

☐ B2. Did you discuss the license or terms for use and / or distribution of any artifacts?
Not applicable. Left blank.

☐ B3. Did you discuss if your use of existing artifact(s) was consistent with their intended use, provided that it was specified? For the artifacts you create, do you specify intended use and whether that is compatible with the original access conditions (in particular, derivatives of data accessed for research purposes should not be used outside of research contexts)?
Not applicable. Left blank.

☐ B4. Did you discuss the steps taken to check whether the data that was collected / used contains any information that names or uniquely identifies individual people or offensive content, and the steps taken to protect / anonymize it?
Not applicable. Left blank.

☐ B5. Did you provide documentation of the artifacts, e.g., coverage of domains, languages, and linguistic phenomena, demographic groups represented, etc.?
Not applicable. Left blank.

☐ B6. Did you report relevant statistics like the number of examples, details of train / test / dev splits, etc. for the data that you used / created? Even for commonly-used benchmark datasets, include the number of examples in train / validation / test splits, as these provide necessary context for a reader to understand experimental results. For example, small differences in accuracy on large test sets may be significant, while on small test sets they may not be.
Not applicable. Left blank.

C ☑ Did you run computational experiments?

Left blank.

☑ C1. Did you report the number of parameters in the models used, the total computational budget (e.g., GPU hours), and computing infrastructure used?
4.1 , 9

The Responsible NLP Checklist used at ACL 2023 is adopted from NAACL 2022, with the addition of a question on AI writing assistance.

☑ C2. Did you discuss the experimental setup, including hyperparameter search and best-found hyperparameter values?
4.1

☑ C3. Did you report descriptive statistics about your results (e.g., error bars around results, summary statistics from sets of experiments), and is it transparent whether you are reporting the max, mean, etc. or just a single run?
7

☑ C4. If you used existing packages (e.g., for preprocessing, for normalization, or for evaluation), did you report the implementation, model, and parameter settings used (e.g., NLTK, Spacy, ROUGE, etc.)?
4.1

D ☒ Did you use human annotators (e.g., crowdworkers) or research with human participants?

Left blank.

☐ D1. Did you report the full text of instructions given to participants, including e.g., screenshots, disclaimers of any risks to participants or annotators, etc.?
Not applicable. Left blank.

☐ D2. Did you report information about how you recruited (e.g., crowdsourcing platform, students) and paid participants, and discuss if such payment is adequate given the participants' demographic (e.g., country of residence)?
Not applicable. Left blank.

☐ D3. Did you discuss whether and how consent was obtained from people whose data you're using/curating? For example, if you collected data via crowdsourcing, did your instructions to crowdworkers explain how the data would be used?
Not applicable. Left blank.

☐ D4. Was the data collection protocol approved (or determined exempt) by an ethics review board?
Not applicable. Left blank.

☐ D5. Did you report the basic demographic and geographic characteristics of the annotator population that is the source of the data?
Not applicable. Left blank.