Efficient Zero-Shot Semantic Parsing with Paraphrasing from Pretrained Language Models

Anonymous ACL submission

Abstract

Building a domain-specific semantic parser with little or no domain-specific training data remains a challenging task. Previous work has shown that crowdsourced paraphrases of synthetic (grammar-generated) utterances can be used to train semantic parsing models for new domains with good results. We investigate whether semantic parsers for new domains can be built with no additional human effort, obtaining paraphrases of grammar-generated utterances from large neural language models, such as Google's T5 and EleutherAI's GPT-J, as an alternative to crowd-sourcing. While our models trained with automated paraphrases generated by pretrained language models do not outperform supervised models trained with similar amounts of human-generated domain-specific data, they perform well in a zero-shot setting, where no domain-specific data is available for a new domain. Additionally, unlike the current state-of-the-art in zero-shot semantic parsing, our approach does not require the use of large transformer-based language models at inference-time. Using the OVERNIGHT dataset, we show that automated paraphrases can be used to train a semantic parsing model that outperforms or is competitive with state-of-the-art-models in the zero-shot setting, while requiring a small fraction of the time and energy costs at inference time.

1 Introduction

Semantic parsing—the task of mapping natural language utterances to logical forms—is an important aspect of language understanding necessary for many applications, such as question answering (Berant et al., 2013; Shen and Lapata, 2007; Yih et al., 2016), querying databases (Zelle and Mooney, 1996), and ontology induction (Poon and Domingos, 2010). Much of the academic work on semantic parsing is based on existing datasets, while semantic parsers for production use are often trained on painstakingly collected and annotated human data. However, when faced with the challenge of creating a semantic parser for a new domain, such as an interface for an equipment repair database, what is the most efficient way to create the required domain-specific training data? This problem is explored by Wang et al. (2015), who propose using crowdsource workers to create natural language versions of grammar-generated English-like canonical utterances that can be deterministically mapped to logical forms in their framework. While this is an effective method for rapidly building a semantic parser for a new domain, it is inherently limited by the time and cost of having humans create the training data. As the domain becomes more complex, the number of possible combinations of logical forms that need to be converted to natural language and paraphrased becomes increasingly large and unwieldy.

In Berant and Liang (2014) and Wang et al. (2015), the authors use a "manageable set of candidate logical forms", as creating a canonical utterance for every possible entity and relation set would be intractable (Berant et al., 2013). While computational tractability is certainly a concern when dealing with larger domains, one of the key limiting factors to the number of canonical utterances one can utilize in the OVERNIGHT framework is the time and cost of having humans create multiple natural language paraphrases of each canonical utterance. Additionally, in an industrial setting, releasing proprietary data to a crowdsourcing platform for annotation may be inadvisable. To mitigate these issues, we investigate the possibility of completely replacing human-generated training data with paraphrases generated using large state-of-the-art transformer language models, namely Google's text-to-text transformer model T5 (Raffel et al., 2020) and EleutherAI's GPT-J (Wang, 2021). We show that human-generated training data can be effectively replaced with paraphrases of grammar-generated canonical utterances using pretrained language models, thus eliminating the need for human-created paraphrase data.

While generating embeddings from large transformer models at inference time has been demonstrated beneficial in semantic parsing (Xu et al., 2020a), we hypothesize that in small, task-oriented semantic parsing domains, running BERT (Devlin et al., 2019) or other large pre-trained language models at inference time may not be necessary, and that adequate performance may be garnered by training smaller neural models on training samples generated from transformer-based paraphrase models. Additionally, when task-oriented semantic parsing models are deployed in real-world use cases, such as reservation booking bots which may handle a large number of requests, the time and energy cost of running large transformer LMs at inference becomes substantial. By generating paraphrase data prior to training, we effectively take advantage of the knowledge contained in these pretrained transformer models without the computational, financial, and environmental cost of running them at inference time.

To test this hypothesis, we investigate the utility of machine-generated paraphrases in training semantic parsers. We propose a straightforward approach consisting of a paraphrase model based on a large pre-trained language model, which is used only prior to model training, and a Bi-LSTM sequence-to-sequence model (Hochreiter and Schmidhuber, 1997; Sutskever et al., 2014) which we run at inference. We find that the use of paraphrases generated automatically with T5 or GPT-J can replace human-generated data entirely with no reduction in accuracy in most domains on the OVERNIGHT dataset. Further, we show that in the zero-shot setting, wherein a semantic parser is trained with no domain-specific training data, we outperform the current state-of-the-art model proposed by Xu et al. (2020b) on the OVERNIGHT dataset at a fraction of the time and energy requirements.

2 Previous work

One of the key challenges faced by developers creating applications that require precise natural language understanding is finding or generating the labeled data necessary to train effective semantic parsers. The problem is exacerbated by the fact that semantic parsing is often quite domain and topic specific, somewhat limiting the benefit that can be derived from of out-of-domain semantic parsing datasets. As pointed out by Su and Yan (2017), different domains often require different predicates and entities; in fact, 30% to 50% of the tokens in each of the eight domains covered by the OVERNIGHT dataset for semantic parsing (Wang et al., 2015) do not occur in any of the other seven included domains. As a result, cross-domain transfer learning in semantic parsing is somewhat limited, especially in small, task-specific domains. While training a single model on multiple domains has been shown an effective means of improving model performance (Herzig and Berant, 2017), this approach still requires domain-specific training data. More recent work (Su and Yan, 2017) trains a cross-domain semantic parser on data from multiple out-of-domain datasets. Given the relatively wide number of semantic parsing datasets available to researchers and industry, we operate under a similar assumption, though we use human-generated out-of-domain paraphrases only in fine-tuning our paraphrase model. In cases where no out-of-domain training data is available for paraphrase model fine-tuning, developers of semantic parsers could create a grammar for the target domain and generate canonical utterances, as proposed in Wang et al. (2015). These canonical utterances could then be paraphrased to create natural-language equivalents using LM-based paraphrasing without fine-tuning to either supplement or completely replace the human-generated data proposed in Wang et al.'s pipeline.

Various approaches have been proposed to create labeled training data for semantic parsers. As described in Wang et al. (2015), training a semantic parser for a new domain consists of the following steps:

- Defining a seed lexicon of entities and properties required in the domain
- Generating a set of combinations of said entities and properties.
- Using a deterministic grammar to generate pseudo-natural language sentences representing each entity-property combination.
- Paraphrasing these pseudo-language forms to create natural language utterances.
- Training a semantic parser to map each natural language utterance to its corresponding logical form.

Several previous works have used paraphrases of machine generated pseudo-language as the basis for training semantic parsers. For example, Berant and Liang (2014) propose converting predicates expressed in the formal language λ-*DCS* (Liang et al., 2013) to *canonical utterances* using a deterministic grammar and entity descriptions from Google's Freebase KnowledgeBase (Google, 2013). These canonical utterances are then matched with natural language utterances in the WEBQUESTIONS dataset (Berant et al., 2013) using a paraphrase association model.

Wang et al. (2015) expand upon Berant and Liang (2014)'s work by using a human-in-the-loop setup, in which Amazon Mechanical Turk workers are tasked with writing paraphrases for machine-generated canonical utterances. By having humans write the paraphrases, Wang et al. (2015) are able to expand on the number of domains for which they train semantic parsers, rather than being limited to those utterances present in WEBQUESTIONS. However, Wang et al.'s approach introduces a new limiting factor in the development of training data - the time and cost of having humans write paraphrases. Wang et al. (2015) use crowd-sourcing to obtain human-generated training data, which allows for relatively fast and efficient collection of training data for supervised models. However, the use of crowd-sourcing introduces another set of limitations to data quality: annotators are not specifically trained in the target task, nor are they necessarily incentivized to generate high-quality data which can be more time-consuming to create (Hsueh et al., 2009). Additionally, even though relatively low compared to other methods of dataset creation such as expert annotation, the cost of crowd-sourcing can become prohibitive, especially when larger amounts of training data are needed.

Supervised models for semantic parsing on the OVERNIGHT dataset, such as Wang et al. (2015), do not utilize grammar-generated canonical utterances during model training; rather these canonical utterances are simply discarded once they have been used as the basis for the creation of human-generated paraphrases. As pointed out by Cao et al. (2020), this is an inefficient use of the available data, as pseudo-language canonical utterances can themselves be used as training data, and can also be utilized to generate paraphrases automatically using paraphrase models.

In order to eliminate the use of human labor in developing training data, Marzoev et al. (2020) propose to tackle semantic parsing as a semantic search problem. However, their results are not competitive with previous work, and require the use of a large general-purpose language model at inference time. Xu et al. (2020b) propose a model which utilizes machine-generated paraphrases of grammar-generated canonical utterances, which can be deterministically mapped to logical form, to replace human-generated data for training semantic parsers, and are able to achieve impressive results. Similarly, Cao et al. (2020) also propose to generate paraphrases of canonical utterances to conduct unsupervised training of a semantic parser for a new domain. Cao et al. (2020) also demonstrate a semi-supervised model which uses machine generated paraphrases of canonical utterances to supplement human-created paraphrases for model training. The results presented by both Xu et al. (2020b) and Cao et al. (2020) on the OVERNIGHT dataset are competitive with state-of-the-art supervised models even with no human-generated data used to train their parsing models. However, like Marzoev et al. (2020), to achieve competitive results, both require the use of a BERT-based encoder during inference to generate contextualized embeddings, a choice we avoid in order to demonstrate the efficacy of LM-generated paraphrases in building smaller, more efficient semantic parsers for small domains. We show that large pre-trained neural models can be leveraged during training to produce much more economical models with competitive accuracy.

3 Methods

In this paper we explore the effects of using automated paraphrases of grammar-generated canonical utterances, which can be deterministically mapped to logical forms, as training data for semantic parsers for small domains. We build and test all models using the OVERNIGHT dataset (Wang et al., 2015), which contains semantic parsing data for eight separate domains. In the OVERNIGHT dataset, each domain is a set of triples $(U_t \in U, C_t \in C, Z_t \in Z)$. Z is a set of logical forms, C is the set of machine-generated canonical utterances, and U is the set of human-generated paraphrases.

We assume a one-to-one mapping $Z \rightarrow C$. Given that each logical form in the OVERNIGHT dataset is deterministicly mapped to a pseudo-language canonical form, our sequence-to-sequence models generate these canonical forms rather than the λ-*DCS* equivalents, as proposed in Su and Yan (2017). Once generated, these canonical forms can be readily converted to a logical form by means of a grammar.

We frame semantic parsing itself as a sequence-to-sequence task, as proposed by Su and Yan (2017). We build a simple Bi-LSTM encoder-decoder model which we train on various amounts of automatically paraphrased data. We intentionally designed models which do not rely on large pre-trained neural models during inference, making our solution far more computationally and economically efficient, and thus more practical for end-user applications. Rather, we use large transformer language models to generate automated paraphrases of the machine-generated canonical utterances in each domain, and these paraphrases are used as training data. By generating paraphrases using large transformer language models prior to training, we are able to harness a portion of the power of these models without the computational cost of running a large model during inference.

We consider several conditions under which LM-generated paraphrases of canonical utterances may be used to replace in-domain human-generated paraphrases. We created models in the following training data conditions:

- Paraphrases generated by T5 without fine-tuning on out-of-domain data (T5)
- Paraphrases generated by T5 with fine-tuning on out-of-domain human-generated data (FINED-TUNED T5)
- Paraphrases generated by GPT-J with out-of-domain human-generated data used as input context (GPT-J).

To generate data for the T5 and FINE-TUNED T5 conditions, we first fined-tuned T5-base for paraphrasing using the PAWS dataset (Zhang et al., 2019), including data from the Quora Question Pairs dataset[1]. In the T5 condition, no further fine-tuning is performed and this model is used directly for paraphrasing canonical utterances. For the FINE-TUNED T5 condition, we hold out one domain as the target semantic parsing domain and further fine-tune for paraphrasing on the remaining 7 domains. This model is then used to generate paraphrases for the held out domain. The process is repeated 8 times, resulting in one model for each domain.

GPT-J paraphrases were obtained using the GPT-J-6B model available through HuggingFace[2]. Because GPT-J is designed to generate continuations of input text, we provide the model with a context consisting canonical utterance and human-generated paraphrase pairs. As with the fine-tuning of T5 described above, paraphrases from GPT-J are generated using *out-of-domain* human-generated paraphrases as input context. We choose a target domain for which to generate paraphrases and then construct the context input for GPT-J by concatenating a canonical-paraphrase pair from each of the non-target domains. These paraphrases are followed by the canonical utterance from the target domain to be paraphrased. GPT-J then generates a paraphrase of the input canonical utterance. No fine-tuning of GPT-J, other than that in-context fine-tuning (Brown et al., 2020) described above, is conducted prior to generation. Appendix A.1 shows a sample of the context provided to GPT-J, the input canonical utterance, and the resulting paraphrase generated by the model.

Paraphrase model fine-tuning is the only aspect of our methodology which relies on *out-of-domain* human-generated data. At no point is *in-domain* human-generated data used in the semantic parsing model development. In total we generate 10 different paraphrase models for the three conditions; one for T5, 8 for FINE-TUNED T5, and one for GPT-J.

When generating paraphrases via T5 and GPT-J, we recognize the fact that generated paraphrases may hinder or improve the performance of the resulting models depending on their quality. As a result, we tested the paraphrase filtering method described in Xu et al. (2020b), but did not find a significant benefit to model performance. Thus, we take no specific steps to filter paraphrases for quality in the present work. However, we consider the number of paraphrases to generate per canonical utterance n, to be a hyperparameter; this allows us to increase the likelihood of generating quality paraphrases while regulating for model performance. We believe that our strong results demonstrate the efficacy of our proposed method.

To evaluate inference-time cost and efficiency of the models we discuss, we use the Experiment Im-

[1] `https://www.quora.com/q/quoradata/First-Quora-Dataset-Release-Question-Pairs`

[2] `https://huggingface.co/EleutherAI/gpt-j-6B`

pact Tracker toolkit from Henderson et al. (2020). This Python toolkit tracks the run time and total power usage (CPU and GPU) of an application and provides an estimate of the CO_{2eq} cost associated with the energy usage.

4 Experiments

We experiment with generating datasets of varying size, ranging from one LM-generated paraphrase per canonical utterance up to 100 paraphrases per canonical utterance. In section 5 we discuss our process for choosing the optimal value of n and discuss the general effect of increasing the number of paraphrases. We note that only canonical utterances contained in the training set are paraphrased. That is, the validation set for a particular domain is the same regardless of the number of paraphrases per training sample we choose to generate.

For each domain in the OVERNIGHT dataset, we train a Bi-LSTM encoder-decoder model to generate pseudo-language canonical forms from input natural language utterances. As the goal of this paper is to explore the effect of using automated paraphrasing in fixed-domain semantic parsing, we train a separate sequence-to-sequence model for each domain trained on LM-generated natural language paraphrases of domain-specific canonical utterances. The parsing models consist of an RNN encoder with two Bi-LSTM layers of 500 units each, and an RNN decoder with global attention, again with two layers of 500 units each. We use a dropout of 0.1. We experimented using pretrained GloVe embeddings (Pennington et al., 2014) but found no statistical improvement in our models. Rather, embeddings are randomly initialized and updated during model training. All models are trained using OpenNMT (Klein et al., 2017). Training and validation sets for each domain were generated by performing a 80/20 split of its official OVERNIGHT training set; where all human utterances in the training split are discarded. Evaluation was conducted using the official OVERNIGHT test set for the target domain, which consists of human utterances only.

5 Choosing the number of examples

To investigate the effect of increasing numbers of example paraphrases on model performance, we compared the accuracy of the resulting models on the OVERNIGHT validation set. Figure 1 shows the average validation accuracy across all domains versus the number of paraphrases per canonical utterance from fine-tuned T5.

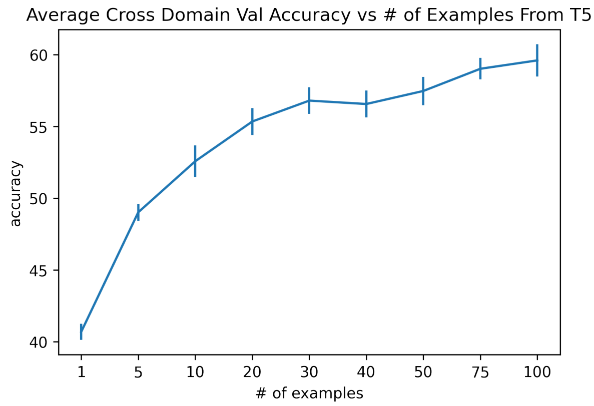

Figure 1: Average cross-domain validation accuracy increases as the number of paraphrases from fine-tuned T5 increases.

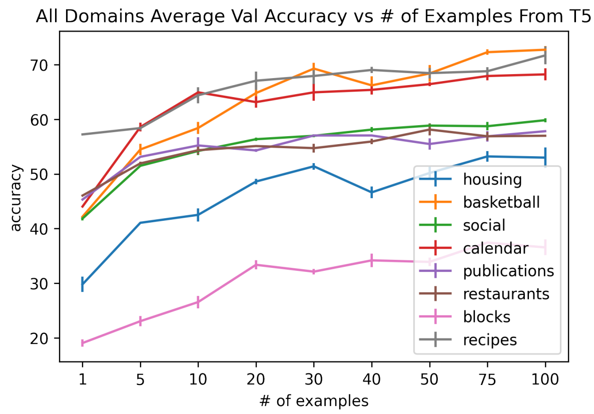

Figure 2: Average validation accuracy vs number of paraphrases from fine-tuned T5 for each domain.

We find that the average cross-domain accuracy generally increases as we include more example paraphrases. This might be the case if only a few domains greatly benefited from increased paraphrasing, however we found that all domains benefit from an increased number of paraphrases. Figure 2 shows how the accuracy for each domain increases as we increase the number of paraphrases. Although all domains see accuracy improvements as we increase the number of paraphrases, not all domains benefit equally. The *Basketball* domain benefits the most, with an improvement over 40% between $n = 1$ and $n = 100$, while the *Restaurants* domains benefits the least with an improvement slightly over 10%.

What causes a domain to be more susceptible to accuracy improvements from increased paraphrasing is unknown. For example, the *Basketball* domain contains 18% more canonical utterances in the training set than the *Restaurants* domain, so it may make sense to see greater relative improvement on the *Basketball* domain compared to *Restaurants* if beginning with more canonical utterances resulted in better performance from paraphrasing. However, the *Housing* domain is approximately half the size of the *Restaurants* domain, but sees an accuracy improvement of 28% compared to *Restaurants*' 10%. After performing this comparison for all domains, the accuracy improvement gained from increasing the number of example paraphrases generated does not seem to be correlated with the number of canonical examples in the domain. We investigated other qualities for each domain (e.g., average utterance length, number of unique utterances, and number of distinct utterances with label overlap) which could possibly affect affinity for paraphrasing, but did not find any conclusive results. See Appendix A.2 for examples of generated paraphrases from each data condition, for each domain.

Regardless of paraphrase quality or relative accuracy improvements between domains, we see that for each domain, generating more paraphrases has an overall positive effect on the resulting semantic parsing model. Both the T5 and FINE-TUNED T5 conditions see similar relative accuracy improvements with increasing numbers of paraphrases. Although this general upward trend in accuracy improvement is promising, it is clear there is a point of diminishing returns. Further, the resources required to generate the paraphrases become prohibitive with an increasing number of paraphrases, as we will discuss in section 7. For this reason we limit our generation to $n = \leq 100$. We leave further investigation into which data features impact the effectiveness of paraphrasing for a particular domain and utterance type to future work.

6 Results

Table 1 shows our results (with varying numbers of example paraphrases) on all domains in the OVERNIGHT dataset alongside results from previous works which do not use in-domain data. We report accuracy on exact match of the output for each sentence. Accurate output is defined as an exact string match between the model output and the corresponding canonical utterance in the test set. Any deviation from the target canonical utterance, however small, is considered inaccurate output. This approach is in keeping with the method used in previous work such as Wang et al. (2015); Marzoev et al. (2020); Xu et al. (2020b). All reported results are the average of five runs of the target condition.

We first compare each data-condition using the maximum number of example paraphrases per canonical utterance. Using 100 paraphrases from FINE-TUNED T5 results in models which have an average cross-domain accuracy of 58.9%. These models outperform equivalent models trained on human-only data on all but two domains. Compared to the current state-of-the-art zero-shot method (Xu et al., 2020b), we achieve a 3.3 percentage point higher cross-domain accuracy.

Next we find that paraphrases generated from T5 result in much less accurate parsing models. With 100 example paraphrases per canonical utterance, the non-fine-tuned condition models achieve an accuracy 15 percentage points lower than those using fine-tuned T5. It is clear that fine-tuning on out-of-domain data with similar sentence structure enables T5 to generate better paraphrases for this task.

Finally, we see models generated using only 10 example paraphrases from GPT-J result in models which outperform those generated with 100 examples from T5 condition by 2.5 percentage points. GPT-J seems to be able to generate much stronger paraphrases than non-fine-tuned T5, as we can see slightly better model accuracy with an order of magnitude less the number of paraphrases, though we should note that GPT-J does have access to a limited amount (10 examples) of similar out-of-domain data in the form of generation prompts. However, when compared to the FINE-TUNED T5 condition, even with the same number of example paraphrases used, the GPT-J condition performs much worse by at least 5 percentage points. We should note that we chose to generate a maximum of 10 example paraphrases from GPT-J due to the significant time and computational cost of running this model, as discussed in Section 7.

As previously discussed, increasing the number of example paraphrases per canonical utterance increases the generated model accuracy on the validation set. Therefore, in an attempt to reduce the total time it takes to produce a model (that is, time spent both on paraphrase generation and model training) one could train a model on a more mod-

condition	Basketball	Blocks	Calendar	Housing	Recipes	Social	Publications	Restaurants	Avg
Marzoev et al. (2020)	47	27	32	36	49	28	34	43	37
Synthetic Only	9.2	14.58	5.59	8.47	11.29	7.26	16.27	21.39	11.76
Human Only	75.96	33.68	49.4	40.74	66.78	64.44	59.01	47.23	54.66
Xu et al. (2020b)	70.1	38.4	58.9	51.9	64.4	47.2	56.5	57.5	55.6
Fine-Tuned T5 (10 ex)	61.38	27.02	58.33	43.7	66.76	54.57	54.24	47.74	51.7
Fine-Tuned T5 (50 ex)	72.43	32.58	56.15	53.04	74.17	59.55	56.77	53.16	57.23
Fine-Tuned T5 (100 ex)	76.21	36.24	57.86	55.56	75.69	60.86	56.52	52.41	58.92
T5 (50 ex)	48.47	33.83	37.62	32.28	60.19	32.92	40.37	43.88	41.2
T5 (100 ex)	55.69	34.74	34.29	35.87	62.04	34.73	45.96	46.99	43.79
GPT-J (1 ex)	42.56	29.7	34.05	27.72	46.06	35.72	36.96	38.86	36.45
GPT-J (10 ex)	60.51	32.83	43.93	38.41	55.93	45.09	44.1	49.58	46.30

Table 1: Accuracy results and comparison to previous work. Our results are an average of five runs and others are copied from cited papers.

est number of a paraphrases with the trade-off of reduced accuracy. We see that even when reducing the number of paraphrases generated, the models generated have competitive accuracy. Training on 50 example paraphrases per canonical utterance from fine-tuned T5 results in models which have a cross-domain accuracy of 57.2%, still slightly higher than both the current state-of-the-art zero-shot models and models trained on human-only data. Similarly, training models on 50 example paraphrases per canonical utterance from T5 or just 1 example paraphrase per canonical utterance from GPT-J results in a cross-domain accuracy of 41.2% and 36.5%, respectively, competitive with Marzoev et al. (2020).

7 Efficiency and Execution Time

W/ GPU	Time (s)	$kgCO_{2eq}$	kWh
Bi-LSTM	102.28	1.09×10^{-3}	4.37×10^{-3}
AutoQA	2898.8	3.95×10^{-2}	0.158
W/O GPU	**Time (s)**	**$kgCO_{2eq}$**	**kWh**
Bi-LSTM	340.78	7.85×10^{-4}	3.13×10^{-3}
AutoQA	5129.33	1.22×10^{-2}	4.88×10^{-2}

Table 2: Average execution time of the Bi-LSTM model and AutoQA with GPU (top) and on CPU only (bottom)

Model	Time (s)	$kgCO_{2eq}$	kWh
T5	0.39	2.59×10^{-6}	1.03×10^{-5}
GPT-J	17.6	4.13×10^{-4}	1.65×10^{-3}

Table 3: Averages per utterance to paraphrase for T5 and GPT-J

In this section we compare the economic and environmental impact of our simple Bi-LSTM encoder-decoder model with the BERT-LSTM model from Xu et al. (2020b) by calculating the execution time and cost of inference on the same dataset. Further, we also compare the cost of paraphrase generation between T5 and GPT-J. As previously mentioned in Section 3, we use the Experiment Impact Tracker toolkit (Henderson et al., 2020) to get accurate benchmarks.

For the comparison of our Bi-LSTM model with the BERT-LSTM model of Xu et al. (2020b), we focus only on the cost accrued during inference time due to the fact that over the lifetime of most deployed neural network models, the cost associated with inference will eventually outweigh the original cost of training (Patterson et al., 2021). We use the publicly available Genie NLP toolkit[3] along with the OVERNIGHT models found on the author's website [4] to compare our work to AutoQA (Xu et al., 2020b). To give a good estimation of the execution time and efficiency of both models we test each on a custom data set which contains the *Basketball* domain test set repeated 100 times (a total of 39100 total utterances). Additionally, we measure and discount the total time and energy cost by the amounts spent loading the model(s) into memory to better capture solely the difference in the cost associated with inference.

Since GenieNLP generates prediction statistics (accuracy, BLEU scores, etc) by default during inference, and OpenNMT does not, we modified the GenieNLP code slightly to omit generating these statistics so the comparisons would be more equitable. Otherwise, both models are run with their default inference parameters.

The experiment was run on a machine with an Intel Xeon ES-2640 v4 CPU @ 2.4GHz, a 12GB

[3] https://github.com/stanford-oval/genienlp

[4] https://wiki.almond.stanford.edu/releases

NVIDIA GTX 1080 Ti GPU, and 64 GB of RAM. We run our experiment twice, once utilizing the GPU and another only using the CPU. Again, reported results are the average of five runs. Table 2 summarizes the results of the experiment.

First, when utilizing both GPU and CPU, we find using our Bi-LSTM encoder-decoder model results in a 28.4x speedup when compared to the AutoQA model on the same dataset. Similarly, our Bi-LSTM model utilizes 2.77% of the estimated $kgCO_{2eq}$ and kWh cost to execute when compared to AutoQA.

When run without using the GPU, we find a 15.1x speedup when using the Bi-LSTM model compared to AutoQA. The difference between the energy consumption of the two models is also reduced, with the Bi-LSTM model using 6.44% of the estimated $kgCO_{2eq}$ and kWh cost to execute when compared to AutoQA.

For the comparison of T5 and GPT-J for paraphrase generation, the model is first loaded into memory and then benchmarked solely on paraphrase generation to focus only on the inference cost of paraphrasing. We select 120 utterances from the *Basketball* training set and they are paraphrased once each. This is repeated 5 times and the results are reported as the average unit divided by 120 (e.g, seconds per utterance). The experiment was run on a machine with an Intel Xeon ES-2620 CPU @ 2.10 GHz, 512 GB of RAM, and an array of 12GB NVIDIA GTX 1080 Ti GPUs. T5 utilized a single GPU and GPT-J was split evenly across two GPUs. Table 3 summarizes the results of our experiment.

We find that generating paraphrases using T5 results in a 45x speedup when compared to paraphrasing the same utterance using GPT-J. Further, T5 requires just 0.63% of the $kgCO_{2eq}$ and kWh cost per utterance used by GPT-J. While we see from the section 6 that GPT-J generated paraphrases can be used to train a semantic parsing model with fewer overall paraphrases than can be done with T5, it's clear this efficiency is paid for in the time and energy cost required to generate them.

8 Conclusion

In this paper we investigate the use of machine-generated paraphrases to replace human-generated paraphrases in the framework initially laid out by Wang et al. (2015). As pointed out by the authors of that paper, they must limit the number of logical forms for which they generate example natural-language utterances in a given domain, as the number of potential logical forms is quite large. However, if we can successfully remove the human-in-the-loop, or at least reduce their role in the process of generating training data, we stand to expand the number of forms which can be covered. Further, the time required and cost of building a semantic parser for a new domain is significantly reduced.

By training a relatively small Bi-LSTM encoder-decoder model with paraphrases generated by a large language model such as T5 and GPT-J, we seek to build an efficient system that benefits from the linguistic and domain-relevant knowledge contained within these models without the need of using a large language model during inference. Our findings that all human-generated data in the OVERNIGHT dataset can be effectively replaced with automatically generated paraphrases without reducing model accuracy in all but two domains is a key finding of this paper.

Further, our model performance on strictly automated paraphrases surpasses the state-of-the-art levels presented in Xu et al. (2020b) and our choice to use simpler parsing models is more practical for end-user applications. We show that large language models can be leveraged during the training phase and their performance gains can be realized with a fraction of the time, energy, and environmental costs associated with deploying them at inference time. Specifically, we show that our relatively small LSTM encoder-decoder model uses roughly 3% of the resources required of the current state-of-the-art model, with an improved overall accuracy.

Finally, we show preliminary results that this method of data generation is generalizable to other large language models, such as GPT-J, where fine-tuning would be impossible due to a lack of similar data with the quantity needed, or infeasible for most users due to the computational resources needed to do so.

In future work, we plan to conduct a detailed analysis of the paraphrases generated by T5 and GPT-J to better understand the types of canonical utterances that these models are most capable of paraphrasing. This will allow us to choose those logical forms, such as highly compositional utterances, that most benefit from human-in-the-loop paraphrasing. We plan to expand this work to new domains for which no training data currently exists to test the effectiveness of our approach in rapidly deploying semantic parsers for new domains.

9 Ethical considerations

The present work is part of an ongoing effort to reduce reliance on large, computationally and environmentally costly language models in NLP research. As demonstrated, our proposed method is able to compete with previous SOTA methods at a fraction of the cost in terms of computational resources and CO_2 emissions. In an NLP environment where ever-increasing language model size seems to be the norm, we strongly believe that harnessing the power of these models in an efficient manner is essential to the long-term sustainability of language processing technology. Additionally, we recognize the potential for bias that exists in current pretrained language models; and by using large pretrained language models to effect the generation of paraphrases, there is opportunity for this bias to propagate through these paraphrases. We are eager to investigate methods of mitigating such bias in our proposed paraphrase models

References

Jonathan Berant, Andrew Chou, Roy Frostig, and Percy Liang. 2013. Semantic parsing on freebase from question-answer pairs. In *Proceedings of the 2013 conference on empirical methods in natural language processing*, pages 1533–1544.

Jonathan Berant and Percy Liang. 2014. Semantic parsing via paraphrasing. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 1415–1425.

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.

Ruisheng Cao, Su Zhu, Chenyu Yang, Chen Liu, Rao Ma, Yanbin Zhao, Lu Chen, and Kai Yu. 2020. Unsupervised dual paraphrasing for two-stage semantic parsing. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, pages 6806–6817.

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.

Google. 2013. Freebase data dumps. `https://developers.google.com/freebase/data`.

Peter Henderson, Jieru Hu, Joshua Romoff, Emma Brunskill, Dan Jurafsky, and Joelle Pineau. 2020. Towards the systematic reporting of the energy and carbon footprints of machine learning.

Jonathan Herzig and Jonathan Berant. 2017. Neural semantic parsing over multiple knowledge-bases. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 623–628, Vancouver, Canada. Association for Computational Linguistics.

Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural Computation*, 9(8):1735–1780.

Pei-Yun Hsueh, Prem Melville, and Vikas Sindhwani. 2009. Data quality from crowdsourcing: a study of annotation selection criteria. In *Proceedings of the NAACL HLT 2009 workshop on active learning for natural language processing*, pages 27–35.

Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. Opennmt: Open-source toolkit for neural machine translation. In *Proceedings of ACL 2017, System Demonstrations*, pages 67–72.

Percy Liang, Michael I Jordan, and Dan Klein. 2013. Learning dependency-based compositional semantics. *Computational Linguistics*, 39(2):389–446.

Alana Marzoev, Samuel Madden, M Frans Kaashoek, Michael Cafarella, and Jacob Andreas. 2020. Unnatural language processing: Bridging the gap between synthetic and natural language data. *arXiv preprint arXiv:2004.13645*.

David Patterson, Joseph Gonzalez, Quoc Le, Chen Liang, Lluis-Miquel Munguia, Daniel Rothchild, David So, Maud Texier, and Jeff Dean. 2021. Carbon emissions and large neural network training. *arXiv preprint arXiv:2104.10350*.

Jeffrey Pennington, Richard Socher, and Christopher D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543.

Hoifung Poon and Pedro Domingos. 2010. Unsupervised ontology induction from text. In *Proceedings of the 48th annual meeting of the Association for Computational Linguistics*, pages 296–305.

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. *Journal of Machine Learning Research*, 21(140):1–67.

Dan Shen and Mirella Lapata. 2007. Using semantic roles to improve question answering. In *Proceedings of the 2007 joint conference on empirical methods in natural language processing and computational natural language learning (EMNLP-CoNLL)*, pages 12–21.

Yu Su and Xifeng Yan. 2017. Cross-domain semantic parsing via paraphrasing. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, pages 1235–1246.

Ilya Sutskever, Oriol Vinyals, and Quoc V. Le. 2014. Sequence to sequence learning with neural networks.

Ben Wang. 2021. Mesh-Transformer-JAX: Model-Parallel Implementation of Transformer Language Model with JAX. https://github.com/kingoflolz/mesh-transformer-jax.

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.

Silei Xu, Giovanni Campagna, Jian Li, and Monica S Lam. 2020a. Schema2qa: High-quality and low-cost q&a agents for the structured web. In *Proceedings of the 29th ACM International Conference on Information & Knowledge Management*, pages 1685–1694.

Silei Xu, Sina Semnani, Giovanni Campagna, and Monica Lam. 2020b. Autoqa: From databases to q&a semantic parsers with only synthetic training data. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 422–434.

Wen-tau Yih, Matthew Richardson, Christopher Meek, Ming-Wei Chang, and Jina Suh. 2016. The value of semantic parse labeling for knowledge base question answering. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, pages 201–206.

John M Zelle and Raymond J Mooney. 1996. Learning to parse database queries using inductive logic programming. In *Proceedings of the national conference on artificial intelligence*, pages 1050–1055.

Yuan Zhang, Jason Baldridge, and Luheng He. 2019. PAWS: Paraphrase adversaries from word scrambling. 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 1298–1308, Minneapolis, Minnesota. Association for Computational Linguistics.

A Appendix

A.1 GPT-J Context

- Original: number of steals (over a season) of player kobe bryant whose number of played games (over a season) is 3
  Paraphrase: how many all season steals did kobe bryant have in 3 games
  ⋮
  Original: block that is right of block that block 1 is left of
  Paraphrase: find me a block that block 1 is to the left of
  Original: number of ingredient
  Paraphrase: how many ingredients are needed
  Original: housing unit whose size is larger than size of 123 sesame street
  Paraphrase:
  - **Generated Paraphrase:**
    - show me apartments larger than 123 sesame street

A.2 Example Paraphrases

- **Canonical**: housing unit whose size is larger than size of 123 sesame street
  - **Human**
    - housing that is bigger than 123 sesame street
    - housing units outsizing 123 sesame street
  - **T5**
    - Housing unit whose size is larger than size of 123 sesame street.
    - housing unit whose size is more than the size of 123 Sesame Street House
  - **Fine-Tuned T5**
    - which housing is larger than the size of 123same street
    - find me all the buildings with a size larger than that of 123
  - **GPT-J**
    - show me apartments larger than 123 sesame street
    - what housing unit is not 123 sesame street
- **Canonical**: person that is friends with student whose end date is at most 2004

- **Human**
  - friend of student whose end date is not after 2004
  - find students friends who ended in 2004 or before
- **T5**
  - Who is friends with a student at least whose end date is 2004
  - The person is a friend of a student whose end date is at most 2004
- **Fine-Tuned T5**
  - who is a friend that is not related to the end date 2004
  - find a pal that has an end date no later than 2004
- **GPT-J**
  - students whose end date is at least the end date of student whose friend is the same person as the person
  - show me students who are friends with students that are still studying

- **Canonical**: season of player kobe bryant
  - **Human**
    - which season was kobe bryant
    - what year did kobe bryant play
  - **T5**
    - Kobe bryant season of player kobe bryant
    - Kobe bryant is the championship player.
  - **Fine-Tuned T5**
    - what is the season of kobe bryant
    - what season does kobe bryant have
  - **GPT-J**
    - what season is player kobe bryant in
    - what season was bryant in

- **Canonical**: recipe whose preparation time is at least cooking time of rice pudding
  - **Human**
    - show me recipes with preparation time equal to or longer than rice pudding
    - show me recipes that have the same or longer preparation time as rice pudding
  - **T5**
    - Recipe whose preparation time is cooking time of rice pudding.
    - The baking time of rice pudding recipe is at least 3 days. The cooking time of a potato is 0 days.
  - **Fine-Tuned T5**
    - whos got a different recipe than rice pudding
    - recipes prepared at least as long as rice pudding
  - **GPT-J**
    - find recipes that involve cooking time greater than or equal to rice pudding preparation time
    - what recipe can be prepared in the same time as rice pudding

- **Canonical**: article that has the most number of author
  - **Human**
    - article with the largest amount of authors
    - what article has the most authors
  - **T5**
    - article with the most authors having the most articles with the most authors having the most articles with the most author having the most articles with the most authors having the most articles with the most author having the least number of articles.
    - The article with the most author(s) has the most author(s)' names.
  - **Fine-Tuned T5**
    - which article belongs to the most people
    - what article has the most number of authors
  - **GPT-J**
    - what article has the most number of authors
    - what article has been written by the most number of authors

- **Canonical**: cuisine that is cuisine of the least number of restaurant
  - **Human**
    - what cuisine is served by the fewest restaurants

- ∗ what cuisine has the least amount of restaurants
  - **T5**
    - ∗ Cuisine that is the cuisine with the least number of restaurants
    - ∗ Cuisine that is cuisine of the lowest number of restaurants
  - **Fine-Tuned T5**
    - ∗ which cuisine is used least in restaurants
    - ∗ what cuisine has the fewest number of restaurants
  - **GPT-J**
    - ∗ find me a cuisine with the least number of restaurants
    - ∗ cuisine with the fewest number of restaurants

- **Canonical**: meeting whose start time is larger than end time of weekly standup
  - **Human**
    - ∗ meetings that start later than the weekly standup meeting
    - ∗ meeting whose start time is after end time of weekly standup
  - **T5**
    - ∗ Meeting whose start time is larger than the end time of a weekly standup meeting
    - ∗ Meeting whose start time is larger than the end time of weekly standups will have the same start time as the first time as the other morning meeting.
  - **Fine-Tuned T5**
    - ∗ which meetings have a start date after the end date of the weekly standup
    - ∗ find me all people who began school after the end date of weekly standup
  - **GPT-J**
    - ∗ what meeting takes longer to start than weekly standup
    - ∗ which meeting starts later than weekly standup

- **Canonical**: block that block 1 is left of and whose length is 3 inches
  - **Human**
    - ∗ what block is to the right of block 1 and has a length of 3 inches
    - ∗ are there any 3inch long blocks to the right of block 1
  - **T5**
    - ∗ Block of which block 1 is left and whose length is 3 inches
    - ∗ Block of block of which block 1 is left and whose length is 3 inches.
  - **Fine-Tuned T5**
    - ∗ which blocks are left on my left with a length of 3 inches
    - ∗ which blocks are 2 inches thick and are left of the blocks
  - **GPT-J**
    - ∗ block 1 that block 1 is left of and whose length is 3 inches
    - ∗ what block is left of and whose length is 3 inches