AutoQA: From Databases To Q&A Semantic Parsers With Only Synthetic Training Data

Silei Xu∗ Sina J. Semnani∗ Giovanni Campagna Monica S. Lam
Computer Science Department
Stanford University
Stanford, CA, USA
{silei,sinaj,gcampagn,lam}@cs.stanford.edu

Abstract
This paper proposes AutoQA, a methodology and toolkit to generate semantic parsers that answer questions on databases, with no manual effort. Given a database schema and its data, AutoQA automatically generates a large set of high-quality questions for training that covers different database operations. AutoQA coaxes the phrasing of questions for the domain out of large pre-trained language models.

AutoQA uses a masked language model to find synonyms of attributes in a table, word masking to determine adjective qualifiers, automatic paraphrasing to find alternative expressions of an attribute in different parts of speech, and a novel filtered auto-paraphraser to generate correct paraphrases for entire sentences.

We apply AutoQA to the Schema2QA dataset and obtain an average logical form accuracy of 70.5% when tested on natural questions, which is only 2.4% lower than a model trained with expert natural language annotations and human paraphrase data. To demonstrate the generality of AutoQA, we also apply it to the Overnight dataset. AutoQA achieves 73.4% answer accuracy, 20% higher than the state-of-the-art zero-shot models and only 5.6% lower than the best model trained with human data.

1 Introduction
Semantic parsing is the task of mapping natural language sentences to executable logical forms. It has received significant attention in question answering systems for structured data (Wang et al., 2015; Zhong et al., 2017; Yu et al., 2018b; Xu et al., 2020). However, training a semantic parser with good accuracy requires a large number of annotated data, which is expensive to acquire. The complexity of logical forms means annotating the data has to be done by an expert which adds to the cost and hinders extending question answering to new databases and domains.

To eliminate the need of annotating data with logical forms, SEMPRE (Wang et al., 2015) proposed the new methodology of first synthesizing questions on the database, then manually paraphrasing them. Recently, Xu et al. (2020) showed that it is possible to achieve high accuracy on realistic user inputs with a comprehensive set of generic, domain-independent question templates. However, it requires a significant manual effort for each domain: the developers must supply how each attribute can be referred to using different parts of speech, and crowdworkers are needed to paraphrase the queries.

Our objective is to eliminate the need for manual effort in building semantic parsers, while achieving comparable accuracy. We hypothesize that, for common domains, the knowledge of how each attribute would be referred to in natural language is implicitly presented in large text corpora and captured by existing pre-trained language models. With that insight, we have developed AutoQA,
a toolkit that (1) automatically annotates the attributes of a database using pre-trained models, (2) uses generic templates to synthesize a large set of complex queries, and (3) uses a novel auto-paraphraser to increase variety of the synthesized data. The data is then used to train a BERT-LSTM model (Xu et al., 2020). The architecture of AutoQA is shown in Fig. 1.

The contributions of this paper are:

• AutoQA, a toolkit that automatically creates a semantic parser that answers questions about a given database. As the parser is trained only with synthetic data, its cost of development is significantly lower than current approaches.
• A novel hybrid algorithm for annotating database attributes: word prediction to find contextual synonyms of the attributes, word masking to identify implicit attributes, and auto-paraphrasing to find alternative expressions in different parts of speech (Section 3).
• A new automatic paraphrasing model, based on BART (Lewis et al., 2019), that can generate natural paraphrases of sentences, with a filter trained with synthetic data to ensure the preservation of the original meaning expressed in a formal language (Section 4).
• The methodology has been tested on the Overnight (Wang et al., 2015) dataset and Schema.org web data (Section 5). On Overnight, AutoQA achieves an average of 57.7% logical form accuracy and 73.4% denotation (answer) accuracy without using the human paraphrases for training, both of which are 20% higher than the state-of-the-art zero-shot models. On five different domains of Schema.org web data, AutoQA achieves an average logical form accuracy of 70.5%, within 3% of models trained with manual annotations and human paraphrases.¹

2 Related Work

Bootstrapping Semantic Parsers. Neural semantic parsing for question-answering is a well-known research topic (Pasupat and Liang, 2015; Wang et al., 2015; Dong and Lapata, 2016; Jia and Liang, 2016; Krishnamurthy et al., 2017; Zhong et al., 2017; Yu et al., 2018b). State of the art methods use a sequence-to-sequence architecture with attention and copying mechanism (Dong and Lapata, 2016; Jia and Liang, 2016) and rely on large datasets. Acquiring such datasets is expensive, and the work must be replicated in every new domain.

Prior work proposed bootstrapping semantic parsers using paraphrasing (Wang et al., 2015). In this technique, a dataset is synthesized using a grammar of natural language, and is paraphrased by crowdworkers to form the training set. Paraphrasing has been applied to datasets for SQL (Zhong et al., 2017), as well as multi-turn dialogue datasets (Shaw et al., 2018).

Genie (Campagna et al., 2019) previously proposed training with large amounts of synthesized and smaller amounts of paraphrased data. Later, Schema2QA (Xu et al., 2020) developed a general template grammar that they found effective for the question answering task on the Web. Both works rely on manual paraphrases and hand-tuned annotations on each database attribute. Annotating the attributes is time-consuming even for domain experts; getting good coverage requires several iterations of refinement on real questions.

A different line of work proposed training with a large multidomain dataset, and then using transfer learning to generalize to new datasets, in a completely zero-shot fashion (Herzig and Berant, 2018a; Chang et al., 2019). Yet, such scenario requires acquiring the multidomain dataset in the first place, and there is a significant gap between the accuracy of training with and without in-domain data (Yu et al., 2018b). Our approach instead is able to synthesize data for the new domain, so the model is exposed to in-domain data while retaining the zero-shot property of no human-annotated data.

Pre-trained Models for Data Augmentation. Previous work showed that pre-trained models are very effective at generalizing natural language knowledge in a zero- and few-shot fashion (Radford et al., 2019; Brown et al., 2020). These models have been used to expand training data for various NLP classification tasks, by fine-tuning the model on a small seed dataset, then using conditioning on the class label to generate more data (Anaby-Tavor et al., 2019; Kumar et al., 2020). Kobayashi (2018) proposed using a bidirectional LSTM-based language model to substitute words that fit the context, conditioning on the class label to prevent augmentation from changing the class label. Wu et al. (2019) used BERT (Devlin et al., 2018) in a similar way, and Hu et al. (2019b) improved upon it by jointly fine-tuning BERT and the classifier.

These approaches rely on an initial dataset with many examples in each class, and therefore are not

¹We will make our code and datasets publicly available.
suitable for semantic parsing, where each logical form has only a few or even just one example. Our method instead does not require any initial data.

**Automatic Paraphrasing for Data Augmentation.** The performance of many NLP tasks can be improved by adding automatically generated paraphrases to their training set. The general approach is to build a paraphrase generation model, usually a neural model (Prakash et al., 2016, Iyyer et al., 2018, Gupta et al., 2017), using general-purpose datasets of paraphrase pairs.

This approach has been applied to various tasks such as sentiment analysis (Iyyer et al., 2018), intent classification (Roy and Grangier, 2019), and span-based question answering (Yu et al., 2018a). Automatic paraphrasing may generate training examples that do not match the original label. Noisy heuristics (Yu et al., 2018a) are not enough for semantic parsing, where paraphrases need to be semantically equivalent in a very strict and domain-dependent sense. We propose a novel filtering approach, and show its effectiveness in reducing the noise of automatic paraphrasing.

### 3 Automatic Annotation

Previous work (Campagna et al., 2019; Xu et al., 2020) showed that high-quality synthetic data is complementary to paraphrase data to train a semantic parser. More diverse and natural synthetic sentences also yield better paraphrases. However, natural common questions are often composed of multiple attributes and simple synthetic strategies might lead to unnatural clunky sentences. For example, in the natural question “Italian restaurants in NY rated 5 stars” which searches for “restaurants whose cuisine is Italian, location is NY, rating value is 5”, three attributes are used.

To generate complex yet natural sentences, Schema2QA (Xu et al., 2020) proposed to use a limited set of about 600 generic templates with grammar rules for different part-of-speech (POS) phrases. Developers can supply domain-specific POS-labeled annotations for each database attribute, which describe how that attribute is used in questions. For example, we can annotate the cuisine attribute to be adjective, location with preposition phrase “in ...”, and rating with passive verb phrase “rated ... stars”. These annotations serve as domain-specific templates with value placeholders, which can be expanded to all possible combinations by the generic templates. A dataset generated by this method is very useful for teaching compositionality to a semantic parser.

Our AutoQA toolkit automatically provides the attribute annotations with the help of pre-trained neural models. Our tool creates unambiguous annotations for all parts of speech, as shown in the examples in Table 1. We first derive a canonical annotation for each attribute from its name with heuristics (Xu et al., 2020), and use a POS tagger to identify its category. We then create alternative annotations with three methods: (1) contextual synonym generation based on BERT, (2) value-driven adjective qualifier identification based on BERT with word masking, and (3) POS-based annotation extraction from auto-paraphrasing.

#### 3.1 Contextual Synonym Generation

Pre-trained language models such as BERT have already embedded knowledge of contextual synonyms. However, when fine-tuning on a lot of data with limited lexical variety, the pre-trained lexical similarity information is often lost. We combat that behavior by explicitly recovering the knowledge of contextual synonyms from the pre-trained model, and including it in the synthesized dataset.

AutoQA uses BERT (Devlin et al., 2018), a pre-trained masked language model (MLM), to discover such contextual synonyms. Using the canonical annotation, AutoQA generates a few short phrases using fixed templates appropriate for the POS category. Each phrase is encoded with the MLM, without any masking. AutoQA then obtains the most likely words from MLM’s output in the positions corresponding to the words of the canonical annotations. Those words are used as new annotations. For example, when given “serves Italian cuisine”, AutoQA generates “serves Italian dishes” (Table 1). “dishes” is not a synonym of “cuisine” in general, but in the context it has the same meaning.

As we generate annotations, which will be used with many different values during synthesis, it is important that we avoid value-specific words. For example, the phrase “with seafood” is meaningful, but “with Italian” is not, so the annotation “with \{value\}” is not appropriate for the cuisine attribute. To avoid generating such invalid annotations, we generate a batch of sentences with different constructs and different values, and only choose synonyms that are common across sentences.
Table 1: Example annotations generated by AutoQA where \{value\} denotes the placeholder for database values.

<table>
<thead>
<tr>
<th>Algorithm</th>
<th>Domain</th>
<th>Attribute</th>
<th>Annotations</th>
<th>POS</th>
<th>Example phrase</th>
</tr>
</thead>
<tbody>
<tr>
<td>Contextual synonym generation</td>
<td>Restaurant</td>
<td>servesCuisine</td>
<td>serves {value} dishes</td>
<td>Verb</td>
<td>restaurant that serves Italian dishes</td>
</tr>
<tr>
<td></td>
<td>Book</td>
<td>award</td>
<td>{value} prize</td>
<td>Noun</td>
<td>books with Nobel prize</td>
</tr>
<tr>
<td></td>
<td>Movie</td>
<td>duration</td>
<td>{value}</td>
<td>Noun</td>
<td>movies with length 2 hours</td>
</tr>
<tr>
<td>Adjective qualifier Identification</td>
<td>Restaurant</td>
<td>servesCuisine</td>
<td>{value} Adjective</td>
<td>Italian restaurant</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Movie</td>
<td>genre</td>
<td>{value} Adjective</td>
<td>Horror movies</td>
<td></td>
</tr>
<tr>
<td>POS-based Annotation Extraction</td>
<td>Book</td>
<td>author</td>
<td>wrote {value} Active verb</td>
<td>who wrote Harry Potter</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Book</td>
<td>author</td>
<td>written by {value} Passive verb</td>
<td>books written by Stephen King</td>
<td></td>
</tr>
</tbody>
</table>

3.2 Adjective Qualifier Identification

In some cases, there is no mention of the attribute of interest in a question at all. Consider the example “Show me an Italian restaurant”; “Italian” is a value of the attribute named “servesCuisine”, and neither “serves” nor “cuisine” is in the utterance.

In such cases, the only information available is the value in the database. We propose a value-driven approach to identify such attributes, using word masking on an MLM. The model is applied to sentences where a [MASK] token is placed before the name of each table. For example, when fed “show me a [MASK] restaurant”, the MLM will predict popular words used before “restaurant”. We then identify the attributes whose values are the predicted words. For example, if the MLM predicts the word “Italian”, “Chinese”, etc., we identify the “servesCuisine” attribute, which includes those values. For the identified attributes, we create adjective annotations of the form \{value\} (Table 1).

3.3 POS-based Annotation Extraction

An attribute may be described using different parts of speech. For example, the “author” attribute can be used in both the verb phrase “wrote” and the passive verb phrase “written by”, as shown in Table 1. To capture more diversity, and to generalize to all combinations with other attributes, we extract annotations from short paraphrases.

We apply the neural paraphrasing model (Section 4) to short canonical sentences for each attribute, to collect alternative phrases. Based on the sentence structure and POS tags in the sentence, we extract phrases to use as natural language annotations. Since the sentences are constructed with one attribute-value pair, it is possible to create a set of common patterns for each part of speech. Paraphrasing phrases for one attribute at a time forces the model to yield more diverse expressions.

AutoQA creates multiple sentences for each attribute with different construct and sample values. Only phrases appearing frequently are added to the final list of annotations. Examples of the annotations obtained by this approach are in Table 1.

3.4 Resolving Conflicts

Word prediction, masking, and paraphrases are imprecise methods and can generate incorrect annotations. Our priority is to eliminate ambiguity: training on nonsensical sentences is acceptable, as such sentences will not appear at test time. Consider a movie domain with both “director” and “creator” attributes. An MLM might generate the annotation “creator” for “director”. To avoid generating such conflicted annotations within the domain, we detect annotations that appear in two or more attributes of the same type in the database. If such an annotation shares the same stem as one attribute name, it is assigned uniquely to that attribute. Otherwise, it is dropped entirely. As we train with data that is synthesized compositionally, we would rather lose a bit of variety than risk introducing ambiguity.

4 Automatic Paraphrasing

Synthetic training data is good for providing coverage with a large number of perfectly annotated sentences. However, even with the 600 generic templates, these artificial sentences are still lacking in naturalness and variety. This is why paraphrasing is critical in semantic parsing. Here we describe how to approximate manual paraphrases with neural paraphrasing models.

4.1 Automatic Paraphrasing and Noise

Using automatic paraphrases in training is challenging. First, paraphrasing models output noisy sentences, partially due to the noise in the existing paraphrasing datasets. We cannot accept para-
phrases that change the meaning of the original sentence, which is represented by the logical form annotation. This noise problem exists even in human paraphrasing; Wang et al. (2015) reports that 17% of the human paraphrases they collected changed the logical form. Second, there is an inherent diversity-noise trade-off when using automatic generation. The more diverse we want to make the outputs, the noisier the model’s output will be. Third, the auto-paraphraser is fed with synthetic sentences, which have a very different distribution compared to the pretraining dataset.

We have found empirically the following ways in which noise is manifested:

- The output is ungrammatical or meaningless.
- The output changes in meaning to a different but valid logical form, or rare words like numbers and proper nouns are changed.
- The model is “distracted” by the input sentence due to limited world knowledge. “I’m looking for the book the dark forest”, is very different from “I’m looking for the book in the dark forest”.
- The model outputs sentence pairs that can be used interchangeably in general, but not in the specific application. For example, “restaurants close to my home” and “restaurants near me” have different target logical forms.
- Automatically-generated annotations are not reviewed by a human to ensure their correctness. An example is the word “grade” instead of “stars” in “hotels with 5 grade”. Further paraphrasing these noisy sentences amplifies the noise.

4.2 Paraphrase Filtering

How do we produce semantically correct paraphrases and yet obtain enough variety to boost the accuracy of the parser? Our approach stands on three observations:

1. Adding training examples that a model already parses correctly (but has not seen during training) can be an effective approach to data augmentation. One manifestation of this idea is self-training (McClosky et al., 2006, He et al., 2019), where a model is used to annotate unlabeled data. We, on the other hand, already know the labels (logical forms) of the paraphrased data and want to verify them. We use a parser to do this verification.

2. Paraphrases of the synthetic dataset are still relatively similar to that set. Thus, a parser trained on synthetic data, which delivers near perfect accuracy for the synthetic data, has a very high accuracy on the paraphrased data as well.

3. Unlike classification tasks, the set of valid logical forms in semantic parsing is so large that outputting the right logical form by chance is very unlikely.

The first observation is the basis of filtering. We feed auto-paraphrased sentences to a parser trained on only synthetic sentences. We accept the sentences as correct paraphrases only if this parser outputs a logical form equal to the original logical form. The synthetic parser is qualified to do this task (second observation). Such sentences can be introduced to train another parser from scratch, which will have a higher accuracy on the natural validation and test sets. Note that this filtering scheme might throw away a portion of correct paraphrases as well, but filtering noisy examples is more important. The third observation ensures that the number of false positives is low.

With this approach to filtering, the first parser (i.e. the filter) can correctly parse the examples present in the synthetic set, e.g. “I am looking for the movies which have Tom Hanks in their actors with the largest count of actors.”, and generalizes to paraphrased sentences like “I’m looking for Tom Hanks movies with the most actors in them.”. After adding these examples to the training set and training, the second parser can generalize to new sentences, e.g. the more natural sentence “What is the Tom Hanks movie with the biggest cast?” As an evidence for the increased generalization, we note that if this process is repeated with the second parser as filter, in practice, it accepts almost twice as many examples as the first filter.

5 Experiments

In this section, we evaluate the effectiveness of our methodology: can a semantic parser created with AutoQA match the performance of human-written data? We evaluate on two different benchmark datasets: the Schema2QA dataset (Xu et al., 2020) and the Overnight dataset (Wang et al., 2015).

5.1 AutoQA Implementation

Template-based Data Synthesis. We leverage the set of 600 generic templates proposed by Xu et al. (2020) to synthesize training data. We found that part of the English grammar was not well captured, and added a few templates, including two
Table 2: Size of Schema2QA dataset and AutoQA generated dataset, with the new domains we added.
Table 3: Test accuracy of AutoQA on the Schema2QA dataset, and comparison with the state of the art.

<table>
<thead>
<tr>
<th>Model</th>
<th>Restaurants</th>
<th>People</th>
<th>Hotels</th>
<th>Books</th>
<th>Movies</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema2QA (Xu et al., 2020)</td>
<td>74</td>
<td>78</td>
<td>65 (transfer)</td>
<td>-</td>
<td>-</td>
<td>-</td>
</tr>
<tr>
<td>Schema2QA w/ updated templates</td>
<td><strong>76.6</strong></td>
<td><strong>78.1</strong></td>
<td>66.5</td>
<td>73.6</td>
<td><strong>69.9</strong></td>
<td><strong>72.9</strong></td>
</tr>
<tr>
<td>Schema2QA w/o manual annotation and paraphrase</td>
<td>39.8</td>
<td>45.4</td>
<td>67.9</td>
<td>61.2</td>
<td>27.3</td>
<td>48.3</td>
</tr>
<tr>
<td>AutoQA</td>
<td>76.6</td>
<td>69.1</td>
<td>72.6</td>
<td>66.1</td>
<td>68.3</td>
<td>70.5</td>
</tr>
</tbody>
</table>

Table 4: Ablation study on Schema2QA development sets. Each “–” line removes only that feature from AutoQA. Adjective qualifier identification has no effect for the People and Hotel domains as no attribute is commonly used as adjective qualifier.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Restaurants</th>
<th>People</th>
<th>Hotels</th>
<th>Books</th>
<th>Movies</th>
<th>Average</th>
</tr>
</thead>
<tbody>
<tr>
<td>Schema2QA w/ updated templates</td>
<td>77.0</td>
<td>75.6</td>
<td>68.9</td>
<td>74.4</td>
<td>73.5</td>
<td>73.9</td>
</tr>
<tr>
<td>Baseline (base canonical, synthetic only)</td>
<td>38.1</td>
<td>41.0</td>
<td>70.1</td>
<td>61.8</td>
<td>27.4</td>
<td>47.7</td>
</tr>
<tr>
<td>AutoQA</td>
<td><strong>78.6</strong></td>
<td><strong>69.6</strong></td>
<td><strong>72.5</strong></td>
<td>67.6</td>
<td><strong>74.9</strong></td>
<td><strong>72.6</strong></td>
</tr>
<tr>
<td>Auto-annotation only</td>
<td>68.8</td>
<td>56.5</td>
<td>71.3</td>
<td>63.3</td>
<td>64.2</td>
<td>64.8</td>
</tr>
<tr>
<td>Auto-paraphrase only (w/ canonical annotation)</td>
<td>54.8</td>
<td>62.9</td>
<td>70.1</td>
<td>66.2</td>
<td>47.9</td>
<td>60.4</td>
</tr>
<tr>
<td>– Contextual synonym generation</td>
<td>70.1</td>
<td>67.3</td>
<td>59.5</td>
<td>70.5</td>
<td>64.7</td>
<td>66.4</td>
</tr>
<tr>
<td>– Adjective qualifier identification</td>
<td>74.1</td>
<td><strong>69.6</strong></td>
<td><strong>72.5</strong></td>
<td>68.1</td>
<td>70.7</td>
<td>71.0</td>
</tr>
<tr>
<td>– POS-based annotation extraction</td>
<td>78.0</td>
<td>62.2</td>
<td>72.2</td>
<td>67.1</td>
<td>67.0</td>
<td>69.3</td>
</tr>
<tr>
<td>– Paraphrase filtering</td>
<td>59.3</td>
<td>55.2</td>
<td>48.0</td>
<td>57.5</td>
<td>64.7</td>
<td>56.9</td>
</tr>
</tbody>
</table>

and annotated them for validation and test. When collecting validation and test data, workers do not see any synthetic data or any values in the database, thus the sentences are not biased by our training data, and contain unseen values. For the purpose of comparison, we also manually annotated the attributes and collected human paraphrase data. Statistics of these datasets are shown in Table 2.

Our metric is logical form accuracy: the query produced by our parser must match the query in the test set exactly. As shown in Table 3, AutoQA achieves an average accuracy of 70.5% in five domains, only 2.4% lower compared to the models trained with manual natural language annotations and human paraphrases. More specifically, AutoQA is 9% and 7.5% lower on people and books domain, but achieves similar or higher accuracy for restaurants, hotels, and movies questions. This result is obtained on test data that are naturally sourced, and not paraphrased. This shows that AutoQA is effective for bootstrapping question answering systems for new domains, without any manual training data.

5.3 Ablation Study

We conduct an ablation study on the development set to evaluate how each part of our methodology contributes to the accuracy. We subtract different components from AutoQA, generate the training data, and run the experiment with the same hyper-parameters. When paraphrase filtering is removed, we use string-matching to remove examples where entities and numbers in the utterance do not match the logical form. Results are shown in Table 4.

Similar to the results on test set, AutoQA reaches the accuracy trained with human paraphrases in three out of five domains. On people and books domains, AutoQA is 6% to 6.8% lower. AutoQA outperforms the baseline trained on synthetic data generated from the canonical annotation in all domains, with an improvement of 40.5% and 37.7% for restaurants and movies. This indicates that AutoQA is an efficient and cost-effective replacement for manual annotation and paraphrasing.

On average, applying only auto-annotation without any paraphrasing achieves 64.8%, which is 7.8% lower than the full AutoQA. Applying only auto-paraphrase based on the heuristically generated canonical annotations obtains 60.5%, and is 12.2% lower than AutoQA. This shows that the two components of AutoQA complement each other to achieve the best performance.

The three auto-annotation algorithms contribute differently for each domain. POS-based annotation extraction contributes the most for the people domain because people attributes are often talked about in various POS, while adjective qualifier prediction helps in restaurants and movies domain, as restaurant cuisine and movie genre are frequently mentioned in adjective form. Overall, the combination of three algorithms always yields the best or close to the best performance.

If auto-paraphrase is used without filtering, the average accuracy drops by 8.3%. This shows that
Table 5: Logical form accuracy (left) and answer accuracy (right) percentage on the Overnight test set. Numbers are copied from the cited papers. We report the numbers for the BL-Att model of Damonte et al. (2019), ZEROSHOT model of Herzig and Berant (2018b), and the Projection model of Marzoev et al. (2020). Herzig and Berant (2018b) do not evaluate on the Basketball domain.

5.4 Applying AutoQA to Overnight

To evaluate if the AutoQA methodology generalizes to different types of databases, logical forms, and templates, we apply AutoQA on the well-known Overnight benchmark. Overnight is a semantic parsing dataset with questions over a knowledge base with very few entities across 8 domains. The dataset was constructed using paraphrasing; all of training, validation and test sets are paraphrased from the same set of synthetic sentences.

We train the BERT-LSTM model using data synthesized from Overnight templates, and apply automatic paraphrasing. Overnight uses a very simple template set to synthesize training examples, with only placeholders for two different parts of speech (verb and noun). We use the standard train/test split and use 20% of the training set for validation.

We evaluate both logical form accuracy and answer accuracy, which checks that the answer is correct in the knowledge base. The BERT-LSTM model outputs only one logical form for each input question, so if that logical form is not syntactically valid, the model obtains 0 accuracy on that test example. Other models benefit from syntax-aware components to mitigate this.

We compare our technique to other approaches that do not use in-domain human data (Table 5). They are either synthetic-only (Marzoev et al., 2020) or use human data from other Overnight domains (Herzig and Berant, 2018b). For reference, we also include the state-of-the-art model (Chen et al., 2018)\(^3\), which uses in-domain human data.

Whereas Schema2QA datasets use naturally sourced evaluation and test data, Overnight evaluates on human paraphrase data. It is not only not as meaningful (Campagna et al., 2019), but also makes the benchmark easier for models trained with the same human paraphrase data. Nonetheless, AutoQA achieves an average logical form accuracy of 57.7% and answer accuracy of 73.4%, which is only 5.6% lower than the state-of-the-art models trained with human paraphrases. Compared to other zero-shot models trained with no in-domain data, AutoQA outperforms the state of the art by 20% on both logical form accuracy and answer accuracy. This shows that by generating diverse and natural paraphrases in domain, AutoQA is able to reach comparable performance with models with human training data, and is much more accurate compared to other zero-shot approaches.

6 Discussion

In this work, we propose AutoQA, a methodology and a toolkit to automatically create a semantic parser given a database. We test AutoQA on two different datasets with different target logical forms and data synthesis templates. On both datasets, AutoQA achieves comparable accuracy to state-of-the-art question answering systems trained with manual attribute annotation and human paraphrases.

AutoQA relies on pre-trained language models and a neural paraphraser trained with a generic out-of-domain dataset to generate training data. We suspect the methodology to be less effective for domains full of jargon. Future work is also needed to handle attributes containing long free-form text, as AutoQA currently only supports standard database training data from a single domain, and do not do transfer-learning from other domains or datasets.

---

\(^3\)This is the best-performing model among those that use without filtering, even a paraphraser with a powerful neural model like BART cannot be used for semantic parsing due to noisy outputs.
operations without reading comprehension.

References

Ateret Anaby-Tavor, Boaz Carmeli, Esther Goldbraich, Amir Kantor, George Kour, Segev Shlomov, Naama Tepper, and Naama Zwerdling. 2019. Not enough data? deep learning to the rescue!


Varun Kumar, Ashutosh Choudhary, and Eunah Cho. 2020. **Data augmentation using pre-trained transformer models.**


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.**


