Understanding Unnatural Questions Improves Reasoning over Text

Xiao-Yu Guo **Yuan-Fang Li** **Gholamreza Haffari**
Faculty of Information Technology, Monash University, Melbourne, Australia
{xiaoyu.guo, yuanfang.li, gholamreza.haffari}@monash.edu

Abstract

Complex question answering (CQA) over raw text is a challenging task. A prominent approach to this task is based on the *programmer-interpreter* framework, where the *programmer* maps the question into a sequence of reasoning actions and the *interpreter* then executes these actions on the raw text. Learning an effective CQA model requires large amounts of human-annotated data, consisting of the ground-truth sequence of reasoning actions, which is time-consuming and expensive to collect at scale. In this paper, we address the challenge of learning a high-quality *programmer* (parser) by projecting natural human-generated questions into unnatural machine-generated questions which are more convenient to parse. We firstly generate synthetic (question, action sequence) pairs by a data generator, and train a semantic parser that associates synthetic questions with their corresponding action sequences. To capture the diversity when applied to natural questions, we learn a projection model to map natural questions into their most similar unnatural questions for which the parser can work well. Without any natural training data, our projection model provides high-quality action sequences for the CQA task. Experimental results show that the QA model trained exclusively with synthetic data outperforms its state-of-the-art counterpart trained on human-labeled data.

1 Introduction

The complex question answering (CQA) task, which requires multi-step, discrete actions to be executed over text to obtain answers, is a challenging task. On the recently released DROP benchmark (Dua et al., 2019), the state-of-the-art method Neural Module Networks (NMNs) (Gupta et al., 2020) learns to interpret each question as a sequence of neural modules, or discrete actions, and execute them to yield the answer. However, the supervised learning of CQA models such as NMNs requires a large amount of annotated (question, action sequence) pairs, which are expensive to acquire and augment. Therefore, the label scarcity problem remains a challenge to the CQA problem.

Motivated by this, we propose a projection model to alleviate the label scarcity challenge by generating synthetic training data. To our knowledge, our model is the first approach using projection to solve the label scarcity problem for the CQA task. The projection model can automatically label large amounts of unlabelled questions with action sequences, so that a CQA model can be trained without natural supervised data. Our method is inspired by the recent "simulation-to-real" transfer approach (Tzeng et al., 2015; Marzoev et al., 2020). As the name suggests, internal knowledge in synthetic data is firstly learned in the "simulation" phase, and then transferred "to-real" circumstances with natural-language utterance using the projection model. In the "simulation" phase, we design an n-gram based generator to produce synthetic training data, i.e. (question, action sequence) pairs which constitute the synthetic dataset. We then train a semantic parser to learn the internal knowledge in this dataset. In the "to-real" phase, we train a projection model that projects each unlabelled natural-language question to a synthetic question and obtain the corresponding action sequence by interpreting the synthetic question using the trained semantic parser. In this way, the internal knowledge is implicitly transferred from "simulation" phase

Proceedings of the 28th International Conference on Computational Linguistics, pages 4949–4955
Barcelona, Spain (Online), December 8-13, 2020

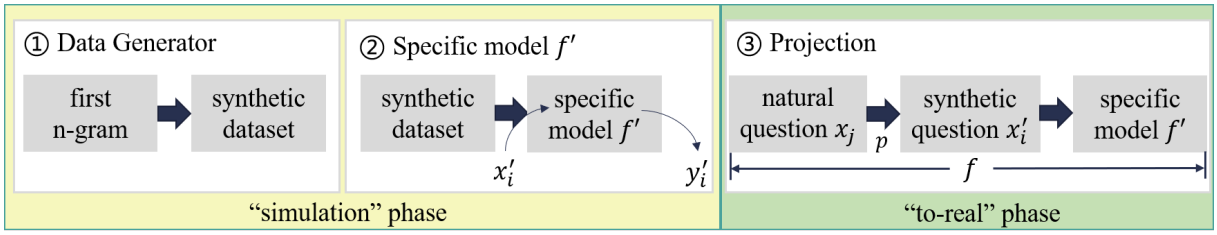

Figure 1: An overview of our proposed model. ① The synthetic dataset is generated by a data generator. ② A specific model f' is trained on the synthetic dataset. ③ A projection model is employed to project natural-language questions to synthetic questions. Note that ① and ② belong to the "simulation" phase, while ③ belongs to the "to-real" phase.

"to-real" phase. With action sequences obtained for the natural questions, an interpreter is employed to execute these action sequences and generate answers consequently.

Experimental results on the challenging DROP dataset demonstrate the effectiveness and practicability of our projection model. Based on the synthetic dataset produced by the data generator, our projection model help NMNs model achieve a 78.3 F1 score on the DROP development dataset. The result indicates the promise of "simulation-to-real" as a development paradigm for NLP problems.

Our contributions are as follows:

- We leverage a projection model to connect synthetic questions with natural-language questions and alleviate the label scarcity problem for the CQA task.
- Our projection model helps generate answers for the CQA task. With high-quality action sequences, the NMNs model achieves higher F1 and Exact Match scores.

2 Background

The complex question answering (CQA) task aims to generate answers for questions that require a thorough understanding of questions and contents such as knowledge base (Hua et al., 2020) or paragraphs. One recent DROP dataset (Dua et al., 2019) fits this case exactly and requires discrete reasoning over contents of paragraphs, which are extracted from Wikipedia. Meanwhile, all questions and corresponding answers are generated and annotated by human workers. The questions in DROP are complex in nature, and are thus especially challenging to existing QA models such as BiDAF (Seo et al., 2017).

Gupta et al. (2020) address this challenge with the end-to-end and fully differentiable Neural Module Networks (NMNs) (Andreas et al., 2016b; Andreas et al., 2016a) and achieves state-of-the-art performance on DROP. The NMNs follows the *programmer-interpreter* structure and consists of two components: a parser (*programmer*) and an *interpreter*. The parser is an encoder-decoder model with attention and learns to interpret a natural-language question into an executable sequence of modules (i.e. actions, and thereafter we use the terms "module" and "action" interchangeably). The interpreter then takes the sequence along with the paragraph as inputs, and predicts the answer after executing the actions one-by-one.

These modules, such as $find$, $filter$ and $count$, are defined to perform independent reasoning over text, numbers and dates. Besides, actions take arguments from questions. For example, the question "How many yards was the first field goal" is interpreted as the first action $find$(field goal), where "field goal" is the argument of the "$find$" action. More details can be found in Appendix A.1.

For realizing the "simulation-to-real" projection model, we define a series of concepts as follows. Formally, our goal is to obtain a semantic parser, i.e. a **general model** $f : \mathcal{Q} \rightarrow \mathcal{T}$ that maps *natural* questions $\mathcal{Q}$ to action sequences $\mathcal{T}$. In this paper, we regard one action sequence as a whole object. Thus, all actions are predicted by semantic parser simultaneously and in order. We approximate this general model by implementing another semantic parser, the **specific model** $f' : \mathcal{Q}' \rightarrow \mathcal{T}$, where $\mathcal{Q}'$ is the set of *synthetic* questions. Note that the model f' is trained on a synthetic dataset $\mathcal{D}' = \{(x'_i, y'_i)\}$, where ($x'_i$,$y'_i$) represents a pair of (synthetic question, action sequence). In order to obtain the action

sequence for a natural question $x_j \in \mathcal{Q}$, we propose a projection model $p : \mathcal{Q} \rightarrow \mathcal{Q}'$, with which we find a synthetic question for each natural question. Therefore, the action sequence for a natural question could be computed using the specific model: $f(x_j) = f'(p(x_j))$. The specific model f' can be trained using a supervised machine learning method on synthetic data only. Therefore, it remains only to find the projection model p.

3 Model

The overall structure of our model is shown in Figure 1. We will firstly introduce an n-gram based data generator in Section 3.1. A cosine-similarity projection model and a classifier-based projection model will follow in Section 3.2. Experimental results demonstrate that the classifier-based project model achieves better performance in both action sequence annotation and question answering task.

3.1 Data Generator

From the existing (question, action sequence) pairs in the training set of the NMNs (Gupta et al., 2020), we firstly summarize a list of first n-grams of questions to provide sufficient coverage of the available action sequences. Note that the "n" in n-gram is an adjustable parameter and there can be **multiple** action sequences for a single n-gram. Some examples are listed in Table 1.

First n-grams	**Synthetic Questions**	**Action Sequences**
how many touchdowns were scored	How many touchdowns were scored?	$count \rightarrow find$
	How many touchdowns were scored by Elam in the first quarter?	$count \rightarrow filter \rightarrow find$
what happened first	What happened first, the crisis or the French Revolution?	$compare\_date\_lesser\_than \rightarrow find\_date, find\_date$
what happened last	What happened last, he died of smallpox or Charles published his Will?	$compare\_date\_greater\_than \rightarrow find\_date, find\_date$

Table 1: Examples of first n-grams, synthetic questions and action sequences. Note that some synthetic questions are not good enough, e.g., the name "Charles" appears after it is first referred by the pronoun "he" in the last example, which is *unnatural* and unfit for language usage.

Secondly, given an n-gram and action sequence, we generate a question with blanks. For example, for the n-gram "what happened first" and the action sequence "$compare\_date\_lesser\_than \rightarrow find\_date, find\_date$", we generate a synthetic question with two blanks to compare two event dates: "what happened first, blank1 or blank2 ?".

Finally, the blanks in each generated question are replaced with names, events, noun phrases, constrained words, etc., from the corresponding paragraph. We extract these various types of entities from natural-language questions and paragraphs using spaCy[1].

3.2 Projection model

A straightforward way to project natural-language questions into synthetic questions is Cosine similarity of question embeddings. We leverage contextualized representations of question words and define the question embedding as the average of all word embeddings in the question. Note that we employ the **bert-base-uncased** model (Devlin et al., 2019) and define the projection model p by:

$$p(x_j) = \arg\min_{x'_i \in \mathcal{X}'} \cos(emb(x_j), emb(x'_i)), \tag{1}$$

where x_j, x'_i represents a natural and synthetic question separately, "cos" represents the Cosine similarity between two vectors. However, Equation 1 requires a large amount of computations, as the entire set of synthetic questions need to be compared for the projection of each natural-language question.

[1] https://spacy.io/

To reduce the time complexity and also improve the performance of the projection model, we enumerate possible action sequences (without arguments) as class labels and treat the projection model as a classification problem. With contextualized representations, we define a new projection model:

$$p(x_j) = classifier(emb(x_j)) = f(x_j). \tag{2}$$

Note that with this classifier-based projection model, we no longer need to employ the specific model f' to further find an action sequence for an input question. Instead, the project model p interprets natural questions as action sequences directly.

4 Experiments

In this section, we will evaluate two projection models in Section 3.2 from two aspects. For one thing, they will be compared with each other, and for another, we employ the classifier-projection model to improve the performance of NMNs (Gupta et al., 2020).

4.1 Dataset and Models

All experiments are performed on the subset of the DROP dataset used in NMNs (Gupta et al., 2020), containing approx. 19,500 (question, answer) pairs for training, 440 for validation and 1,700 for testing. 2,420 questions in the training set and all questions in the validation set have been manually annotated with ground-truth action sequences, which are used to train the NMNs model, which we denote as "original" below. We evaluate two variants of our method based on the training data that is used to train our classifier-based projection model: (1) "synthetic", where the projection model is trained on the 2,420 synthetic questions generated by the data generator in Section 3.1, and (2) "natural", where the projection model is trained on the 2,420 natural questions in the original dataset. Once trained, the projection model is applied to the remaining questions in the training set without action sequences to provide additional training data for NMNs.

4.2 Results

By learning models in Section 3.2, we employ the Cosine projection model on the synthetic dataset and find that it achieves an accuracy of 83.2% on the validation set. Meanwhile, we trained two distinct classifiers using only synthetic dataset or natural dataset, with which we gain 93.2% and 96.1% respectively. These results show that the synthetic dataset produced by the data generator is high-quality enough and the projection model trained on it is comparable to the one trained on natural dataset.

To further evaluate the projection model, we provide NMNs with classifier-based projection results to evaluate the performance on the downstream CQA task on the DROP dasatet (Dua et al., 2019). Concretely, we evaluate our "simulation-to-real" approach in two settings: "synthetic" and "natural", where two classifiers (Section 3.2) are trained on the synthetic dataset and natural dataset respectively. We compare our methods with the original NMNs model that is trained on the NMNs dataset. We denote this baseline method "original".

Methods	**original (baseline)**	**synthetic**	**natural**
F1	77.4	78.3	79.1
EM	74.0	74.9	75.9

Table 2: F1 and EM scores for NMNs trained on different datasets.

In Table 2, we report F1 and Exact Match (EM) scores for the CQA task. As shown in Table 2, compared with the original model, "synthetic" NMNs achieves a higher F1 and EM scores using action sequences generated by our projection model. Moreover, we find the quality of synthetic data can be further improved, as the projection model trained on the natural dataset achieves better performance. There are two main reasons to support the fact that our projection model can help understand complex questions. One is that, our projection model provide more supervised (question, action sequence) pairs

with the parser. Meanwhile, the action sequences produced by the projection model are more accurate than the generated sequences by parser. See details in A.2.

5 Conclusion

In this paper, we propose a projection model that only employs synthetic data to develop supervisions for real-world data. Experimental results show that our approach can be equivalent to supervised learning on natural dataset in performance. In addition, with projection results, we employ the NMNs model to solve complex question answering problem and generate answers for the DROP dataset. Higher F1, Exact Match scores demonstrate that our projection model can help improve the performance of the downstream CQA task and provide a good reference to relevant works.

References

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016a. Learning to compose neural networks for question answering. In *Proceedings of the NAACL HLT 2016*, pages 1545–1554.

Jacob Andreas, Marcus Rohrbach, Trevor Darrell, and Dan Klein. 2016b. Neural module networks. In *Proceedings of the CVPR 2016*, pages 39–48.

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 NAACL HLT 2019*, pages 4171–4186.

Dheeru Dua, Yizhong Wang, Pradeep Dasigi, Gabriel Stanovsky, Sameer Singh, and Matt Gardner. 2019. DROP: A reading comprehension benchmark requiring discrete reasoning over paragraphs. In *Proceedings of the NAACL HLT 2019*, pages 2368–2378.

Nitish Gupta, Kevin Lin, Dan Roth, Sameer Singh, and Matt Gardner. 2020. Neural module networks for reasoning over text. In *Proceeding of the ICLR 2020*.

Yuncheng Hua, Yuan-Fang Li, Gholamreza Haffari, Guilin Qi, and Wei Wu. 2020. Retrieve, program, repeat: Complex knowledge base question answering via alternate meta-learning. In *Proceedings of the IJCAI 2020*, pages 3679–3686.

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

Min Joon Seo, Aniruddha Kembhavi, Ali Farhadi, and Hannaneh Hajishirzi. 2017. Bidirectional attention flow for machine comprehension. In *Proceeding of the ICLR 2017*.

Eric Tzeng, Coline Devin, Judy Hoffman, Chelsea Finn, Xingchao Peng, Sergey Levine, Kate Saenko, and Trevor Darrell. 2015. Towards adapting deep visuomotor representations from simulated to real environments. *arXiv:1511.07111*.

A Appendix

A.1 NMNs model overview

Gupta et al. (2020) propose a Neural Module Networks (NMNs) model to solve the complex question answering problem. Containing a parser and an interpreter, NMNs have a more interpretable structure as shown in Figure 2. Note that the parser and the interpreter are jointly learned with auxiliary supervision in the training period.

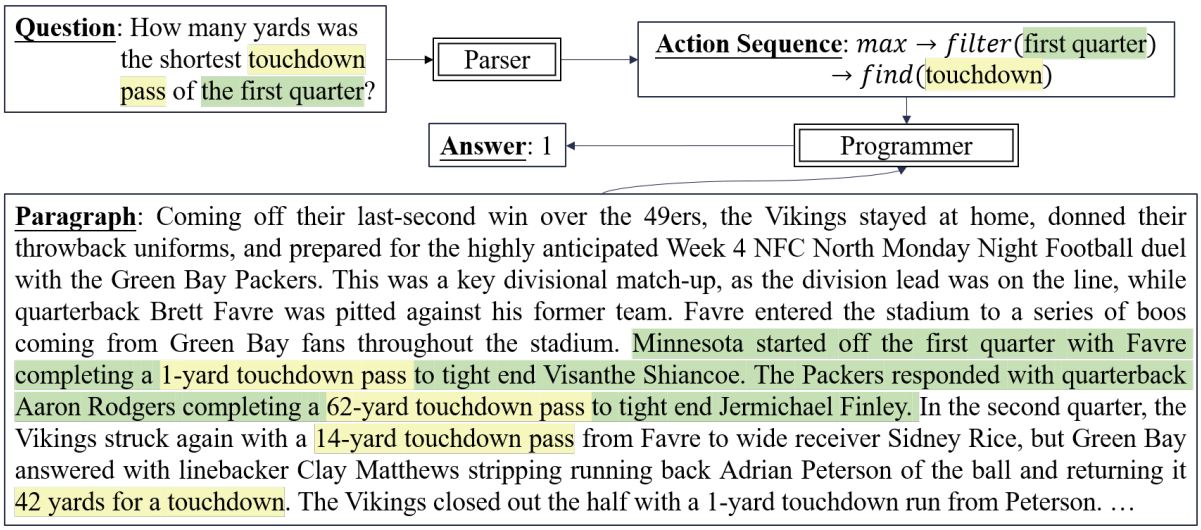

Figure 2: NMNs model architecture.

As Figure 2 shows, NMNs takes as inputs the question and the paragraph. The encoder-decoder based parser firstly interprets the question into an executable action sequence. The action-based interpreter then executes this sequence against the corresponding paragraph to produce the final answer. By calculating attention matrix, all actions independently achieve reasoning over raw text or the outputs from other actions. For example, the $find$(touchdown) action finds all the "touchdown pass" in the paragraph and assigns larger attention weights to all words in yellow background. Then $filter$(first quarter) filters "touchdown pass" belong to the first quarter by producing an attention mask over the output of $find$ action. In Table 3, we list some examples of questions, answers and the corresponding action sequences.

Questions	**Action Sequences**	**Answers**
How many touchdowns did the Giants score in the fourth quarter?	$count \rightarrow filter \rightarrow find$	2
Who kicked the most field goals?	$relocate \rightarrow find$	Rackers
Who threw the longest touchdown pass of the first quarter?	$relocate \rightarrow max \rightarrow filter \rightarrow find$	Aaron Rodgers
How many yards was the longest touchdown reception?	$max \rightarrow find$	14 yards
Which happened earlier, the formation of the United Nations or the dissolution of the Soviet Union?	$compare\_date\_lesser\_than \rightarrow find\_date, find\_date$	formation of the United Nations
How many years after the formation of the United Nations was the Universal Declaration of Human Rights adopted?	$year\_difference \rightarrow find\_date, find\_date$	3 years

Table 3: Examples for questions, answers and action sequences.

A.2 Examples from distinct models

Table 4 lists (question, action sequence) pairs generated by different projection models and compares their impact on the final answer. All questions are selected from the DROP validation dataset, and action sequences are either generated by the parser ("original") or generated by our projection models. As can be seen, our projection methods are able to generate action sequences that lead to correct answers. Apparently, the action sequences from baseline sometimes are erroneous, as shown in the first question, which does not lead to the correct answer.

Question	**Method**	**Action Sequences**	**F1**
How many years did it take for the Allies to take five towns from the Dutch?	original	$year\_diff \rightarrow find, find$	0.0
	synthetic	$year\_diff\_\boldsymbol{single\_event} \rightarrow find\_date$	1.0
	natural	$year\_diff\_\boldsymbol{single\_event} \rightarrow find\_date$	1.0
Which happened first, the second Kandyan War, or Sri Lankan independence?	original	$find\_span \rightarrow compare\_date\_lesser\_than \rightarrow find\_date, find\_date$	0.0
	synthetic	$find\_span \rightarrow compare\_date\_lesser\_than \rightarrow find\_date, find\_date$	0.0
	natural	$find\_span \rightarrow compare\_date\_\boldsymbol{greater}\_than \rightarrow find\_date, find\_date$	0.55
How many years was between the oil crisis and the energy crisis?	original	$year\_diff\_single\_event \rightarrow find\_date$	0.0
	synthetic	$year\_diff\_single\_event \rightarrow find\_date$	0.0
	natural	$year\_diff \rightarrow find\_date$	1.0

Table 4: Different (question, action sequence) pairs obtained when training on different datasets.