Evaluating Semantic Parsing against a Simple Web-based Question Answering Model

Alon Talmor  
Tel-Aviv University  
alontalmor@mail.tau.ac.il

Mor Geva  
Tel-Aviv University  
morgeva@mail.tau.ac.il

Jonathan Berant  
Tel-Aviv University  
joberant@cs.tau.ac.il

Abstract

Semantic parsing shines at analyzing complex natural language that involves composition and computation over multiple pieces of evidence. However, datasets for semantic parsing contain many factoid questions that can be answered from a single web document. In this paper, we propose to evaluate semantic parsing-based question answering models by comparing them to a question answering baseline that queries the web and extracts the answer only from web snippets, without access to the target knowledge-base. We investigate this approach on COMPLEXQUESTIONS, a dataset designed to focus on compositional language, and find that our model obtains reasonable performance (∼35 F₁ compared to 41 F₁ of state-of-the-art). We find in our analysis that our model performs well on complex questions involving conjunctions, but struggles on questions that involve relation composition and superlatives.

1 Introduction

Question answering (QA) has witnessed a surge of interest in recent years (Hill et al., 2015; Yang et al., 2015; Pasupat and Liang, 2015; Chen et al., 2016; Joshi et al., 2017), as it is one of the prominent tests for natural language understanding. QA can be coarsely divided into semantic parsing-based QA, where a question is translated into a logical form that is executed against a knowledge-base (Zelle and Mooney, 1996; Zettlemoyer and Collins, 2005; Liang et al., 2011; Kwiatkowski et al., 2013; Reddy et al., 2014; Berant and Liang, 2015), and unstructured QA, where a question is answered directly from some relevant text (Voorhees and Tice, 2000; Hermann et al., 2015; Hewlett et al., 2016; Kadlec et al., 2016; Seo et al., 2016).

In semantic parsing, background knowledge has already been compiled into a knowledge-base (KB), and thus the challenge is in interpreting the question, which may contain compositional constructions (“What is the second-highest mountain in Europe?”) or computations (“What is the difference in population between France and Germany?”). In unstructured QA, the model needs to also interpret the language of a document, and thus most datasets focus on matching the question against the document and extracting the answer from some local context, such as a sentence or a paragraph (Onishi et al., 2016; Rajpurkar et al., 2016; Yang et al., 2015).

Since semantic parsing models excel at handling complex linguistic constructions and reasoning over multiple facts, a natural way to examine whether a benchmark indeed requires modeling these properties, is to train an unstructured QA model, and check if it under-performs compared to semantic parsing models. If questions can be answered by examining local contexts only, then the use of a knowledge-base is perhaps unnecessary. However, to the best of our knowledge, only models that utilize the KB have been evaluated on common semantic parsing benchmarks.

The goal of this paper is to bridge this evaluation gap. We develop a simple log-linear model, in the spirit of traditional web-based QA systems (Kwok et al., 2001; Brill et al., 2002), that answers questions by querying the web and extracting the answer from returned web snippets. Thus, our evaluation scheme is suitable for semantic parsing benchmarks in which the knowledge required for answering questions is covered by the web (in contrast with virtual assistants for which the knowledge is specific to an application).
We test this model on **COMPLEXQUESTIONS** (Bao et al., 2016), a dataset designed to require more compositionality compared to earlier datasets, such as **WEBQUESTIONS** (Berant et al., 2013) and **SIMPLEQUESTIONS** (Bordes et al., 2015). We find that a simple QA model, despite having no access to the target KB, performs reasonably well on this dataset ($\sim 35$ F₁ compared to the state-of-the-art of 41 F₁). Moreover, for the subset of questions for which the right answer can be found in one of the web snippets, we outperform the semantic parser (51.9 F₁ vs. 48.5 F₁). We analyze results for different types of compositionality and find that superlatives and relation composition constructions are challenging for a web-based QA system, while conjunctions and events with multiple arguments are easier.

An important insight is that semantic parsers must overcome the mismatch between natural language and formal language. Consequently, language that can be easily matched against the web may become challenging to express in logical form. For example, the word “wife” is an atomic binary relation in natural language, but expressed with a complex binary $\lambda x.\lambda y.\text{Spouse}(x,y)$ ∧ Gender$(x,\text{Female})$ in knowledge-bases. Thus, some of the complexity of understanding natural language is removed when working with a natural language representation.

To conclude, we propose to evaluate the extent to which semantic parsing-based QA benchmarks require compositionality by comparing semantic parsing models to a baseline that extracts the answer from short web snippets. We obtain reasonable performance on **COMPLEXQUESTIONS**, and analyze the types of compositionality that are challenging for a web-based QA model. To ensure reproducibility, we release our dataset, which attaches to each example from **COMPLEXQUESTIONS** the top-100 retrieved web snippets.¹

### 2 Problem Setting and Dataset

Given a training set of triples $\{(q^{(i)}, R^{(i)}, a^{(i)})\}_{i=1}^{N}$, where $q^{(i)}$ is a question, $R^{(i)}$ is a web result set, and $a^{(i)}$ is the answer, our goal is to learn a model that produces an answer $a$ for a new question-result set pair $(q, R)$. A web result set $R$ consists of $K(=100)$ web snippets, where each snippet $s_i$ has a title and a text fragment. An example for a training example is provided in Figure 1.

Semantic parsing-based QA datasets contain question-answer pairs alongside a background KB. To convert such datasets to our setup, we run the question $q$ against Google’s search engine and scrape the top-$K$ web snippets. We use only the web snippets and ignore any boxes or other information returned (see Figure 1 and the full dataset in the supplementary material).

**Compositionality** We argue that if a dataset truly requires a compositional model, then it should be difficult to tackle with methods that only match the question against short web snippets. This is since it is unlikely to integrate all necessary pieces of evidence from the snippets.

We convert **COMPLEXQUESTIONS** into the aforementioned format, and manually analyze the types of compositionality that occur on 100 random training examples. Table 1 provides an example for each of the question types we found:

- **SIMPLE**: an application of a single binary relation on a single entity.
- **FILTER**: a question where the semantic type of the answer is mentioned (“tv shows” in Table 1).
- **N-ARY**: A question about a single event that involves more than one entity (“juni” and “spy kids 4” in Table 1).
- **CONJUNCTION**: A question whose answer is the conjunction of more than one binary relation in the question.

---

¹Data can be downloaded from [https://worksheets.codalab.org/worksheets/0x91d77db37e0a4bbaeb37b8972f4784f/](https://worksheets.codalab.org/worksheets/0x91d77db37e0a4bbaeb37b8972f4784f/)
Table 1: An example for each compositionality type and the proportion of examples in 100 random examples. A question can fall into multiple types, and thus the sum exceeds 100%.

| Type       | Example                                                                 | %  |
|------------|------------------------------------------------------------------------|----|
| SIMPLE     | “who has gone out with cornelis de graeff”                           | 17%|
| FILTER     | “which tv shows has wayne roastad starred in”                         | 18%|
| N-ARY      | “who pleased juni jay kids 4?”                                       | 51%|
| CONJ.      | “what has queen latifah starred in that doug mcenery directed”        | 10%|
| COMPOS.    | “who was the grandson of king david’s father?”                       | 7% |
| SUPERL.    | “who is the richest sports woman?”                                   | 9% |
| OTHER      | “what is the name george lopez on the show?”                         | 8% |

Our model comprises two parts. First, we extract a set of answer candidates, $\mathcal{A}$, from the web result set. Then, we train a log-linear model that outputs a distribution over the candidates in $\mathcal{A}$, and is used at test time to find the most probable answers.

**Candidate Extraction** We extract all 1-grams, 2-grams, 3-grams and 4-grams (lowercased) that appear in $R$, yielding roughly 5,000 candidates per question. We then discard any candidate that fully appears in the question itself, and define $\mathcal{A}$ to be the top-$K$ candidates based on their tf-idf score, where term frequency is computed on all the snippets in $R$, and inverse document frequency is computed on a large external corpus.

**Candidate Ranking** We define a log-linear model over the candidates in $\mathcal{A}$:

$$p_\theta(a \mid q, R) = \frac{\exp(\phi(q, R, a)^\top \theta)}{\sum_{a' \in A} \exp(\phi(q, R, a')^\top \theta)},$$

where $\theta \in \mathbb{R}^d$ are learned parameters, and $\phi(\cdot) \in \mathbb{R}^d$ is a feature function. We train our model by maximizing the regularized conditional log-likelihood objective $\sum_{i=1}^N \log p_\theta(a^{(i)} \mid q^{(i)}, R^{(i)}) + \lambda \cdot \|\theta\|^2_2$. At test time, we return the most probable answers based on $p_\theta(a \mid q, R)$ (details in Section 4). While semantic parsers generally return a set, in COMPLEXQUESTIONS 87% of the answers are a singleton set.

**Features** A candidate span $a$ often has multiple mentions in the result set $R$. Therefore, our feature function $\phi(\cdot)$ computes the average of the features extracted from each mention. The main information sources used are the match between the candidate answer itself and the question (top of Table 2) and the match between the context of a candidate answer in a specific mention and the question (bottom of Table 2), as well as the Google rank in which the mention appeared.

Lexicalized features are useful for our task, but the number of training examples is too small to train a fully lexicalized model. Therefore, we define lexicalized features over the 50 most common non-stop words in COMPLEXQUESTIONS. Last, our context features are defined in a 6-word window around the candidate answer mention, where the feature value decays exponentially as the distance from the candidate answer mention grows. Overall, we compute a total of 892 features over the dataset.

**4 Experiments** COMPLEXQUESTIONS contains 1,300 training examples and 800 test examples. We performed 5 random 70/30 splits of the training set for development. We computed POS tags and named entities with Stanford CoreNLP (Manning et al., 2014). We did not employ any co-reference resolution tool in this work. If after candidate extraction, we do not find the gold answer in the top-$K$ candidates, we discard the example, resulting in a training set of 856 examples.

We compare our model, WebQA, to STAGG (Yih et al., 2015) and CompQ (Bao et al., 2016), which are to the best of our knowledge the highest performing semantic parsing models on both COMPLEXQUESTIONS and WEBQUESTIONS. For these systems, we only report test $F_1$ numbers that are provided in the original papers, as we do not have access to the code or predictions. We evaluate models by computing average $F_1$, the official evaluation metric defined for COMPLEXQUESTIONS. This measure computes the $F_1$
Table 2: Features templates used to extract features from each answer candidate mention \(\alpha_m\). Cosine similarity is computed with pre-trained GloVe embeddings (Pennington et al., 2014). The definition of common words and weighting by distance is in the body of the paper.

between the set of answers returned by the system and the set of gold answers, and averages across questions. To allow WEBQA to return a set rather than a single answer, we return the most probable answer \(\alpha^*\) as well as any answer \(\alpha\) such that \((\phi(q, R, \alpha^*))^T \theta - \phi(q, R, \alpha)^T \theta) < 0.5\). We also compute precision@1 and Mean Reciprocal Rank (MRR) for WEBQA, since we have a ranking over answers. To compute metrics we lowercase the gold and predicted spans and perform exact string match.

Table 3 presents the results of our evaluation. WEBQA obtained 32.6 \(F_1\) (33.5 p@1, 42.4 MRR) compared to 40.9 \(F_1\) of COMPQ. Our candidate extraction step finds the correct answer in the top-\(K\) candidates in 65.9% of development examples and 62.7% of test examples. Thus, our test \(F_1\) on examples for which candidate extraction succeeded (WEBQA-SUBSET) is 51.9 (53.4 p@1, 67.5 MRR).

We were able to indirectly compare WEBQA-SUBSET to COMPQ: Bao et al. (2016) graciously provided us with the predictions of COMPQ when it was trained on COMPLEXQUESTIONS, WEBQUESTIONS, and SIMPLEQUESTIONS. In this setup, COMPQ obtained 42.2 \(F_1\) on the test set (compared to 40.9 \(F_1\), when training on COMPLEXQUESTIONS only, as we do). Restricting the predictions to the subset for which candidate extraction succeeded, the \(F_1\) of COMPQ-SUBSET is 48.5, which is 3.4 \(F_1\) points lower than WEBQA-SUBSET, which was trained on less data.

Not using a KB, results in a considerable disadvantage for WEBQA. KB entities have normalized descriptions, and the answers have been annotated according to those descriptions. We, conversely, find answers on the web and often predict a correct answer, but get penalized due to small string differences. E.g., for “what is the longest river in China?” we answer “yangtze river”, while the gold answer is “yangtze”. To quantify this effect we manually annotated all 258 examples in the first random development set split, and determined whether string matching failed, and we actually returned the gold answer.\(^2\) This improved performance from 53.6 \(F_1\) to 56.6 \(F_1\) (on examples that passed candidate extraction). Further normalizing gold and predicted entities, such that “Hillary Clinton” and “Hillary Rodham Clinton” are unified, improved \(F_1\) to 57.3 \(F_1\). Extrapolating this to the test set would result in an \(F_1\) of 34.4 (WEBQA-EXTRAPOL in Table 3) and 34.9, respectively.

Last, to determine the contribution of each feature template, we performed ablation tests and we present the five feature templates that resulted in the largest drop to performance on the development set in Table 4. Note that TF-IDF is by far the most impactful feature, leading to a large drop of 12 points in performance. This shows the importance of using the redundancy of the web for our QA system.

Analysis To understand the success of WEBQA on different compositionality types, we manu-
Figure 2: Proportion of examples that passed or failed candidate extraction for each compositionality type, as well as average $F_1$ for each compositionality type. COMPOSITION and SUPERLATIVE questions are difficult for WEBQA.

Table 4: Feature ablation results. The five features that lead to largest drop in performance are displayed.

| Feature Template | $F_1$  | $\Delta$ |
|------------------|--------|--------|
| WEBQA            | 53.6   |        |
| - MAX-NE         | 51.8   | -1.8   |
| - NE+COMMON      | 51.8   | -1.8   |
| - GOOGLE RANK    | 51.4   | -2.2   |
| - IN QUEST       | 50.1   | -3.5   |
| - TF-IDF         | 41.5   | -12    |

5 Related Work

Our model WEBQA performs QA using web snippets, similar to traditional QA systems like MULDER (Kwok et al., 2001) and AskMSR (Brill et al., 2002). However, it enjoys the advances in commercial search engines of the last decade, and uses a simple log-linear model, which has become standard in Natural Language Processing.

Similar to this work, Yao et al. (2014) analyzed a semantic parsing benchmark with a simple QA system. However, they employed a semantic parser that is limited to applying a single binary relation on a single entity, while we develop a QA system that does not use the target KB at all.

Last, in parallel to this work Chen et al. (2017) evaluated an unstructured QA system against semantic parsing benchmarks. However, their focus was on examining the contributions of multitask learning and distant supervision to training rather than to compare to state-of-the-art semantic parsers.

6 Conclusion

We propose in this paper to evaluate semantic parsing-based QA systems by comparing them to a web-based QA baseline. We evaluate such a QA system on COMPLEXQUESTIONS and find that it obtains reasonable performance. We analyze performance and find that COMPOSITION and SUPERLATIVE questions are challenging for a web-based QA system, while CONJUNCTION and N-ARY questions can often be handled by our QA model.

5 Related Work

Our model WEBQA performs QA using web snippets, similar to traditional QA systems like MULDER (Kwok et al., 2001) and AskMSR (Brill et al., 2002). However, it enjoys the advances in commercial search engines of the last decade, and uses a simple log-linear model, which has become standard in Natural Language Processing.

Similar to this work, Yao et al. (2014) analyzed a semantic parsing benchmark with a simple QA system. However, they employed a semantic parser that is limited to applying a single binary relation on a single entity, while we develop a QA system that does not use the target KB at all.

Last, in parallel to this work Chen et al. (2017) evaluated an unstructured QA system against semantic parsing benchmarks. However, their focus was on examining the contributions of multitask learning and distant supervision to training rather than to compare to state-of-the-art semantic parsers.

6 Conclusion

We propose in this paper to evaluate semantic parsing-based QA systems by comparing them to a web-based QA baseline. We evaluate such a QA system on COMPLEXQUESTIONS and find that it obtains reasonable performance. We analyze performance and find that COMPOSITION and SUPERLATIVE questions are challenging for a web-based QA system, while CONJUNCTION and N-ARY questions can often be handled by our QA model.

Acknowledgments

We thank Junwei Bao for providing us with the test predictions of his system. We thank the anonymous reviewers for their constructive feedback. This work was partially supported by the Israel Science Foundation, grant 942/16.

References

J. Bao, N. Duan, Z. Yan, M. Zhou, and T. Zhao. 2016. Constraint-based question answering with knowl-
edge graph. In *International Conference on Computational Linguistics (COLING)*.

J. Berant, A. Chou, R. Frostig, and P. Liang. 2013. Semantic parsing on Freebase from question-answer pairs. In *Empirical Methods in Natural Language Processing (EMNLP)*.

J. Berant and P. Liang. 2015. Imitation learning of agenda-based semantic parsers. *Transactions of the Association for Computational Linguistics (TACL)* 3:545–558.

K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. 2008. Freebase: a collaboratively created graph database for structuring human knowledge. In *International Conference on Management of Data (SIGMOD)*. pages 1247–1250.

A. Bordes, N. Usunier, S. Chopra, and J. Weston. 2015. Large-scale simple question answering with memory networks. *arXiv preprint arXiv:1506.02075* .

E. Brill, S. Dumais, and M. Banko. 2002. An analysis of the AskMSR question-answering system. In *Association for Computational Linguistics (ACL)*. pages 257–264.

D. Chen, J. Bolton, and C. D. Manning. 2016. A thorough examination of the CNN / Daily Mail reading comprehension task. In *Association for Computational Linguistics (ACL)*.

Danqi Chen, Adam Fisch, Jason Weston, and Antoine Bordes. 2017. Reading Wikipedia to answer open-domain questions. In *Association for Computational Linguistics (ACL)*.

K. M. Hermann, T. Koish, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman, and P. Blunsom. 2015. Teaching machines to read and comprehend. In *Advances in Neural Information Processing Systems (NIPS)*.

D. Hewlett, A. Lacoste, L. Jones, I. Polosukhin, A. Fandrianto, J. Han, M. Kelcey, and D. Berthelot. 2016. Wikireading: A novel large-scale language understanding task over Wikipedia. In *Association for Computational Linguistics (ACL)*.

F. Hill, A. Bordes, S. Chopra, and J. Weston. 2015. The goldilocks principle: Reading children’s books with explicit memory representations. In *International Conference on Learning Representations (ICLR)*.

Mandar Joshi, Eunsol Choi, Daniel S Weld, and Luke Zettlemoyer. 2017. Triviaqa: A large scale distantly supervised challenge dataset for reading comprehension. *arXiv preprint arXiv:1705.03551* .

R. Kadlec, M. Schmid, O. Bajgar, and J. Kleinendienst. 2016. Text understanding with the attention sum reader network. In *Association for Computational Linguistics (ACL)*.

T. Kwiatkowski, E. Choi, Y. Artzi, and L. Zettlemoyer. 2013. Scaling semantic parsers with on-the-fly ontology matching. In *Empirical Methods in Natural Language Processing (EMNLP)*.

C. Kwok, O. Etzioni, and D. S. Weld. 2001. Scaling question answering to the web. *ACM Transactions on Information Systems (TOIS)* 19:242–262.

P. Liang, M. I. Jordan, and D. Klein. 2011. Learning dependency-based compositional semantics. In *Association for Computational Linguistics (ACL)*. pages 590–599.

C. D. Manning, M. Surdeanu, J. Bauer, J. Finkel, S. J. Bethard, and D. McClosky. 2014. The stanford coreNLP natural language processing toolkit. In *ACL system demonstrations*.

T. Onishi, H. Wang, M. Bansal, K. Gimpel, and D. McAllester. 2016. Whodid what: A large-scale person-centered cloze dataset. In *Empirical Methods in Natural Language Processing (EMNLP)*.

P. Pasupat and P. Liang. 2015. Compositional semantic parsing on semi-structured tables. In *Association for Computational Linguistics (ACL)*.

J. Pennington, R. Socher, and C. D. Manning. 2014. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*.

P. Rajpurkar, J. Zhang, K. Lopyrev, and P. Liang. 2016. Squad: 100,000+ questions for machine comprehension of text. In *Empirical Methods in Natural Language Processing (EMNLP)*.

S. Reddy, M. Lapata, and M. Steedman. 2014. Large-scale semantic parsing without question-answer pairs. *Transactions of the Association for Computational Linguistics (TACL)* 2(10):377–392.

M. Seo, A. Kembhavi, A. Farhadi, and H. Hajishirzi. 2016. Bidirectional attention flow for machine comprehension. *arXiv* .

E. M. Voorhees and D. M. Tice. 2000. Building a question answering test collection. In *ACM Special Interest Group on Information Retrieval (SIGIR)*. pages 200–207.

Y. Yang, W. Yih, and C. Meek. 2015. WikiQA: A challenge dataset for open-domain question answering. In *Empirical Methods in Natural Language Processing (EMNLP)*. pages 2013–2018.

X. Yao, J. Berant, and B. Van-Durme. 2014. Freebase QA: Information extraction or semantic parsing. In *Workshop on Semantic parsing*.

W. Yih, M. Chang, X. He, and J. Gao. 2015. Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Association for Computational Linguistics (ACL)*.
M. Zelle and R. J. Mooney. 1996. Learning to parse database queries using inductive logic programming. In Association for the Advancement of Artificial Intelligence (AAAI). pages 1050–1055.

L. S. Zettlemoyer and M. Collins. 2005. Learning to map sentences to logical form: Structured classification with probabilistic categorial grammars. In Uncertainty in Artificial Intelligence (UAI). pages 658–666.