WEBQUESTIONSSP v1.0

Contact Persons: Matt Richardson (mattri@microsoft.com), Scott Wen-tau Yih (scottyih@microsoft.com)

1 Overview

This document includes the usage, format, and other specifics about the WEBQUESTIONSSP dataset. The dataset is being released as part of the paper "The Value of Semantic Parse Labeling for Knowledge Base Question Answering" [1], in which we evaluated the value of gathering semantic parses, vs. answers, for a set of questions that originally comes from WEBQUESTIONS [2]. If you use this dataset, please cite our paper:

```
@InProceedings{YihRichardsonMeekChangSuh:ACL2016:WebQSP,
  author    = {Yih, Wen-tau and Richardson, Matthew and Meek, Christopher and Chang, Ming-Wei and
Suh, Jina},
  title     = {The Value of Semantic Parse Labeling for Knowledge Base Question Answering},
  booktitle = {Proceedings of the 54th Annual Meeting of the Association for Computational Lin-
guistics},
  month     = {August},
  year      = {2016},
  address   = {Berlin, Germany},
  publisher = {Association for Computational Linguistics}
}
```

We believe the data is useful not only for research on question answering and semantic parsing, but also for other tasks such as entity linking (we have, for most questions, an entity link between an entity in the question and an entity in Freebase).

2 Usage

The dataset consists of four files: WebQSP.[test|train].[partial|_].json. The train/test split is the same as WEBQUESTIONS, and every question in WEBQUESTIONS is in WEBQUESTIONSSP (and vice-versa). The train and test sets are additionally split into the primary dataset (WebQSP.train.json, WebQSP.test.json) and additional "partial" questions for which a valid parse could not be formulated or where the question itself is bad or needs a descriptive answer. See the labeling instructions for details on what is considered a bad or descriptive question. The split criteria is: the question is included in the primary dataset if the question itself is labeled as "good", and the parse is labeled as "complete". We are including the partial questions because they still contain potentially useful information. For example, many are still labeled with the primary entity and relation chain, but are incomplete due to being unable to add a necessary constraint. Thus, the partial questions could still be useful when, for example, learning to map a question to the relevant core inferential (relation) chain.

Evaluation on the test set should be reserved for computing final published numbers. We encourage splitting the training set into train and development sets, or using cross-validation, when running experiments. For final numbers, please use the included "eval.py" script, which computes the average precision, average recall, average of per-question-F1.

2.1 Generating your answers for evaluation

The evaluation script compares your answer set with the answer set contained in the data. We generated the answer sets for each question by executing the SPARQL query (included in the data) using OpenLink Virtuoso Open-Source Edition 7.1.0. To ensure the answer sets match, use the following formatting guidelines:

- The SPARQL query should be authored so that when it returns literals, it only returns those with no language tag, or the "en" language tag.
- If the answer is a URI, remove the http://rdf.freebase.com/ns/ prefix. The result should start with "m." or "g.".
- For literals, the answer is just the literal value (e.g., 22), as opposed to the NTriple-formatted string found in the Freebase dump (e.g., "22"@en)
- If the answer is a literal, and the datatype is http://www.w3.org/2001/XMLSchema#*fragment* (for some *fragment*), then convert the value into our standardized datetime strings as follows:

Fragment	Format	Example (for April 10, 1923)
date	YYYY-MM-DD	1923-04-10
dateTime	YYYY-MM-DD	1923-04-10
gYearMonth	YYYY-MM	1923-04
gYear	YYYY	1923

2.2 Questions with relative time

Many questions ask about a fact relative to the time the question is asked, typically with regard to "now". This can be done implicitly ("who does Michael Vick play for?") or explicitly ("who are the two current senators of Illinois?"). For such purposes, we take "now" to be one day after the date of the Freebase dump (which is indicated in the JSON file). Specifically, we are using the dump from 2015-08-09, so "now" is 2015-08-10. Note that this date is spread throughout the file, so if you change the date of the KB, you should be sure and change this date throughout the dataset as well.

3 STAGED QUERY GRAPH

Every complete parse contains a SPARQL query that was used to generate the answer set that answers the question. But about 97% of the questions are answerable using a restricted language, introduced in [3], called a *staged query graph*. For these, the answer can be found by starting from a "topic entity" in the question, following one or more relation links, and constraining the answer set with constraints. For example, for the question *what is the name of justin bieber brother*, we can start at the topic entity *Justin Beiber* (m.06w2sn5), follow the sibling relation chain (*people.person.sibling_s*, *people.sibling_relationship.sibling*) to get all of his siblings, and then add the constraint that any answers must additionally have the relation *people.person.gender* with value "Male".

The staged query graph is provided for all questions that can be answered in the following subset of the language: the inferential chain is either length 1 or 2, and if the length is 2 then the path goes through a CVT node in the first step. The staged query graph is automatically converted into a SPARQL query. For the remaining 3% of the questions which do not satisfy this restriction, the SPARQL is manually authored (indicated by a "#MANUAL" prefix).

4 DATA FORMAT

WebQuestionsSP is formatted in JSON format, with the following schema. See the next section for specifics about the data.

Dataset	**Contains the entire WEBQUESTIONSSP dataset**
string Version	Version of the WEBQUESTIONSSP dataset
string FreebaseVersion	Version of Freebase used to compute answer sets (the date of the Freebase dump)
Question[] Questions	The entire set of questions in the dataset
Question	**A question along with all of its parses**
string QuestionId	Question Id
string RawQuestion	Original question from WEBQUESTIONS
string ProcessedQuestion	Question with some basic processing (remove trailing question mark, tokenization of 's)
Parse[] Parses	One or more semantic parse annotations for the question
Parse	**One semantic parse annotation**
string ParseId	Parse Id
int AnnotatorId	Id of the annotator who generated this parse
ParseComment AnnotatorComment	Comment fields from the annotator
string Sparql	Sparql-formatted query to answer the question. Typically, this is automatically generated from the rest of the fields of this Parse (TopicEntityMid, InferentialChain, Constraints, and Order). For those that didn't fit into this scheme, they were authored by hand, indicated by the prefix "#MANUAL". Null if there is no parse.
string TopicEntityMid	The MID of the primary topic entity of the question. Null if there is no primary entity or it does not exist in Freebase.
string TopicEntityName	Name (type.object.name) of the TopicEntityMid (just for convenience). Null if TopicEntityMid is null.
string PotentialTopicEntityMention	The mention (substring of the question) that corresponds to the topic entity. Null if no entity or none suggested when annotating.
string[] InferentialChain	The chain of Freebase relations leading from the topic entity to the answers. Null if there was no chain (e.g., annotator couldn't find a relation for this question).
Constraint[] Constraints	Set of constraints applied to the query. Empty array if there are no constraints
OrderConstraint Order	Specifies an ordering and selection of the answer set (e.g., for questions like "what are the top 5…" or "which state has the largest…". Null if there is no order constraint needed.
TemporalSemantics Time	Additional information if there are time-oriented constraints in the parse. Null if there are no time constraints
Answer[] Answers	null if the query was not executed. Empty array if the query returns the empty set.
Constraint	**Constraints to restrict the answer set as needed to exactly answer the question**
int SourceNodeIndex	Index of the point along the inferential chain to consider the set of source nodes for the constraint. For example, "0" means the source

	nodes are anything found by following InferentialChain[0] from the TopicEntityMid.
string NodePredicate	Relation to follow from the source nodes. Null means use the source node itself.
OperatorType Operator	{*Equal*, *NotEqual*, *LessThan*, *GreaterThan*, *LessOrEqual*, *GreaterOrEqual*, *Exist*, *NotExist*}. Operator to compare the result of SourceNode->NodePredicate to the Argument
ArgumentDataType ArgumentType	{*Value*, *Entity*}. The type of the argument
ValueDataType ValueType	{*String*, *Number*, *DateTime*}. The type of the value, if ArgumentType is Value.
string Argument	The argument: either a MID if the ArgumentType is Entity, or an appropriately-formatted value if ArgumentType is Value
string EntityName	The entity name if the argument is an entity. Just for convenience.
TemporalSemantics	**Extra information about the temporal semantics of the question.**
bool IsRelativeToNow	Is the time relative to "now" (vs. absolute)
string Start	Start of the time period. If absolute, then a date string formatted as YYYY-MM-DD. If relative, then same format but with each field optionally prefixed by a "-" to indicate years, months, or days before (vs. after) "now". For example, 0000-00-00 means "now", and -0001-00-00 means one year ago.
string End	End of the time period, formatted like "Start".
int[] AssociatedConstraints	If we were able to implement the temporal semantics in actual constraints on Freebase, these are the indices of the constraints in the Constraints array that correspond to the temporal constraints described by this TemporalSemantics.
string PotentialTimeMention	The mention (substring) in the question that caused the temporal semantics. Empty if it has an implicit "now" like "who is the president of the United States".
ParseComment	**Comments about the question and parse from this annotator**
ParseQuality ParseQuality	{Complete, Partial} Is the parse complete, meaning the SPARQL is valid and represents the question being asked. If the SPARQL is not manually authored, this also implies all of the other fields of Parse are valid and filled in.
QuestionQuality QuestionQuality	{Good, Bad, BetterAnsweredByDescription} Is this question good ("what country is Orlando in"), bad ("ho last won the Superbowl"), bad ("which kardashians are having babies?), or better answered with a full description ("what is new york giants"). See the annotation guidelines for more explanation of what is considered "bad" or needing a description.
ParseConfidence Confidence	How well the parse matches the question. See annotation guidelines for more details.
string FreeFormComment	A freeform text comment left by the annotator when annotating this question. Sometimes the first character indicates a special sta-

	tus (marked by the annotator) as described in the labeling instructions. There is also sometimes structure in the comment due to concatenating multiple comment fields, post-processing, and also when the Sparql had to be manually entered, it is indicated with a "!Manual" prefix.
OrderConstraint	**Specifies a subset of the answer set according to sorting (e.g., min, max, top-k, …)**
int SourceNodeIndex	Same as Constraint.SourceNodeIndex
string NodePredicate	Same as Constraint.NodePredicate
ValueDataType ValueType	Same as Constraint.ValueType
SortOrder SortOrder	{Ascending, Descending} The direction in which the items should be sorted
int Start	The position of the first item to output. 0-based
int Count	Number of items to return (>0)
Answer	**One of the answers returned by the SPARQL query**
ArgumentDataType AnswerType	{Value, Entity}. The type of the argument
string AnswerArgument	The answer. A MID if AnswerType is Entity, and a formatted value string if AnswerType is Value.
string EntityName	The name of the entity if the answer is a MID.

5 Additional Data Specifics

5.1 Partially-annotated parses

Some of the questions are unanswerable using Freebase. For these, we still provide partially-annotated parses for the question, as some of the fields could still be useful for training. The annotator progressed through annotating first a primary topic *entity*, then a *relation* or sequence of relations from that entity, and finally *constraints* and *order* (labeled at the same time). If the parse is partial due to an inability to select an appropriate entity, relation, constraint, or order, then the entries up to that point in the annotation process are valid, and those after will be empty. For example, if the annotator could not find a *relation* to answer the question, the TopicEntity will still be filled in, but the constraints and order will be empty.

5.2 Entity names

When entity names are provided (Parse.TopicEntityName, Constraint.EntityName, Answer.EntityName), the names come from following the type.object.name relation from the entity. We provide only the @en (English) literal, if it exists. If the @en literal does not exist, then we provide all literals. The names are simply for convenience; they are not used in the SPARQL query or for evaluation.

5.3 SPARQL generation

When the SPARQL is automatically generated from the staged query graph, we found that comparing two dates needed to be done by subtracting the two and comparing to 0 (e.g., time1 - time2 < 0). Comparing the dates directly (e.g., time1 < time2) resulted in inconsistent results.

The Constraints and OrderConstraints specify the data type when it's a value type so that the SPARQL query can correctly specify the datatype. For example, it's important to specify that two fields are being compared as numbers rather than strings when doing an orderby.

5.4 TemporalSemantics field

The TemporalSemantics (TS) annotation was added later as an augmentation to the dataset. Time constraints are so common in the questions that we felt it was worth calling them out specifically and describing the time period they describe. The SPARQL generation does not use the TS field. Rather, SPARQL generation uses the constraints field, and the TS was added later and points to the constraints that it describes.

Many questions cannot be answered because of time constraints. For example, "what was the currency in Russia ten years ago" cannot be answered using Freebase because Freebase records only the current currency for any given country.

To the best of our ability, every parse that is incomplete because of needing time annotation was given a TS, except:

- WebQTest-486, which would need TS, but TS would need to encode the date of the revolution, which would require another query, and the relation in question (form of government) doesn't have dates anyway.
- WebQTrn-170, where it is unclear what time period they mean.
- Questions which have a missing entity or relation.

If the parse is "partial" and it has a TS with empty "AssociatedConstraints", then the inability to constrain on time is the only issue that makes the parse partial instead of complete. For example, a question about when something because a country's currency cannot be answered, so we add a TS but leave the AssociatedConstraints empty because the TS cannot be grounded into real constraints.

- Except WebQTrn-2404 which has an empty AssociatedConstraints but is marked complete because of having a manually-authored SPARQL.

6 References

[1] W. Yih, M. Richardson, C. Meek, M. Chang & J. Suh. *The Value of Semantic Parse Labeling for Knowledge Base Question Answering*. In ACL-2016.

[2] J. Berant, A. Chou, R. Frostig & P. Liang. *Semantic parsing on Freebase from question-answer pairs*. In EMNLP-2013.

[3] W. Yih, M. Chang, X. He & J. Gao. *Semantic Parsing via Staged Query Graph Generation: Question Answering with Knowledge Base*. In ACL-IJCNLP-2015.