Machine Learning Nanodegree: Capstone Proposal

Finbar Good

November 8, 2018

1 Domain Background

Since Google coined the term “knowledge graph”, there has been an increasing interest in the types of systems that fall under this broad term[1]. Despite the interest - or perhaps because of the lure of the name behind it - the term has been applied widely and inconsistently. For the purposes of this proposal, we can settle on this short definition: any store of facts represented as a graph. And the sub-class of these graphs we are specifically interested in are those that make use of the RDF standard.

RDF (Resource Description Framework) graphs [2] represent facts as a triple of IRIs or literal values. A triple consists of three parts:

- a subject: a thing or class, referred to using an IRI
- a predicate: a property or relationship of the subject
- an object: either the value subject’s property, or the target of relationship

When the content of a graph is specified with RDF, the subjects and objects form the nodes and the predicates form the edges. Because the predicates are directed - going from subject to object - the graph is directed.

SPARQL is a query language designed for accessing RDF data, and was elevated by the W3C to the recommended language for doing so [3]. The following is an example of a very simple query in SPARQL:

```
SELECT ?creator
WHERE { ?creator imbdo:created imdbr:the_wire }
```

In this example, we are looking for the entity (or entities) that created another entity, imdbr:the_wire. There are superficial similarities with SQL, but they work in very distinct ways. What they have in common is, from the point of view of a lay user, a steep learning curve; only specialists tend to have the skills to create queries in either language.

2 Problem Statement

An area of research that has emerged in the past decade has been to broaden the accessibility of knowledge graphs through parsing of natural language queries (NLQ) to extract an answer from a knowledge graph (for example [4]). Some of the more recent efforts in the more general topic of question answering with a knowledge base have looked at using neural networks to solve the problem (for example [5]).

The challenge with accessing the data in knowledge graphs using RDF is the query language, SPARQL. For many casual or non-technical users it is an insurmountable barrier. To broaden the reach of such knowledge graphs requires the automated translation of what casual users can produce, natural language queries (NLQ), into something knowledge graphs can parse, SPARQL.

The project I am proposing is to attempt to implement a neural network to carry out automated translation of NLQs to SPARQL queries. The training will be focussed on a narrow set of simple questions that can be expressed with a limited number of SPARQL templates (queries with placeholders for entities and predicates that are bound at runtime) - however the plan is to test their performance on question types outside the training / testing datasets.

3 Datasets and Inputs

I will be using the LC-QuAD dataset [6]. It was created for the QALD (Question Answering over Linked Data) initiative [7], a series of annual challenges to translate NLQ into SPARQL (or the correct answer to the NLQ). The LC-QuAD dataset consists of 5000 questions and their corresponding SPARQL queries. The questions were generated using the following workflow:

1. Manually create query templates and a natural language equivalent template
2. Extract a list of entities
3. Manually create a whitelist of predicates
4. For each entity, extract subgraph centred on the entity from DBpedia, extending 2 hops away
5. Generate a query from each fact in these subgraphs, restricted to the predicate whitelist
6. The populated template is then mapped to the natural language equivalent
7. Humans review the final result, paraphrasing and/or correcting the results

I will then train the neural network on going in the reverse direction.

Taking an example from [6]:

Template

```
SELECT DISTINCT ?url
WHERE {
    ?x e_in_to_e_in_out e_in_out .
    ?x e_in_to_e ?uri
}
```

Query

```
SELECT DISTINCT ?url
WHERE {
    ?x dbp:league dbr:Turkish_Handball_Super_League .
    ?x dbp:mascot ?uri
}
```

Normalised Natural Question Template

```
What is the `<mascot>` of the `<handball team>` whose `<league>` is `<Turkish
Handball Super League>`?
```

Corrected Question

```
What are the mascots of the teams participating in the turkish handball super
league?
```

4 Solution Statement

For this proposed project I will be taking the same approach as described in [8], which is to use neural machine translation to predicate SPARQL from NLQ.

The solution will:

- tokenise the NLQ and SPARQL datasets
- train a seq3seq neural net on the training subset to predict SPARQL queries from the NLQ
- use the BLEU metric as the prediction metric [9]
- compare the metric across dimensions of (1) number of each query type in the training set (2) number of training epochs

5 Benchmark Model

I will be using the model described in [8] as the benchmark. It defines the approach I will use, but uses a different dataset. Both datasets were constructed from DBpedia, binding entities found in DBPedia to question-query templates. The paper shows a number of BLEU scores, one for each enhancement the team applied to their model or dataset pre-processing.

6 Evaluation Metrics

The BLEU score measures the degree of match between a generated sentence and the reference sentence. This makes it useful to measure success of a translation, in our case from natural language to SPARQL query. Values range from 1 (perfect match) to 0 (perfect mismatch). Roughly, it is a count of the number of distinct n-gram matches found in the reference sentence, normalised by word count.

To use an example given in [9], we have the generated sentence:

```
It is a guide to action which ensures that the military always obeys the commands
of the party
```

If we compare it to this reference sentence:

```
It is a guide to action that ensures that the military will forever heed Party
commands
```

we see that 8 of the bigrams in the generated sentence are found in the reference sentence. The bigram BLEU score is the number of matches divided by the number of bigrams generated i.e. 8 / 17 = 0.47.

Shorter n-gram scores score precision, longer ones score fluency.

I will train on different n-gram BLEU and compare the results. As SPARQL queries are not "fault tolerant", we will need high scores for n > 1.

7 Project Design

The following is an outline of the main activities planned, roughly in the order they will be carried out:

7.1 Construct the dataset pre-processing:

- the source JSON format needs to be separated into the NLQ and SPARQL sets
- punctuation needs to be removed
- the SPARQL queries need to be encoded (it consists of special characters and urls as well as SPARQL keywords)

7.2 Construct the neural network model:

This will be a sequence-to-sequence model, encoding the NLQs, and decoding into SPARQL. I expect the model will use embedding layers and possibly RNN or LSTM layers (the alternatives will be tried out).

The source and target vocabularies also have to be determined, and whether to use all tokens found or whether to truncate these lists.

7.3 Test with small subset of the dataset

This phase is for debugging any issues with the model.

7.4 Analyse dataset for natural subsets

See if there are any subsets that e.g. differ in query type. It was noted in the paper describing the model we are using as reference, that inclusion of queries with multiple entity placeholders had a significant impact on the BLEU score.

7.5 Train / test / validate

7.6 Alter hyper-parameters

For example, number of epochs and optimiser.

7.7 Calculate, compare and visualise results

The BLEU score will be visualised so as to show any drivers discovered that make the model successful / unsuccessful e.g. number of epochs.

If the BLEU score is surprisingly low, then a re-evaluation of the model and dataset preparation may need to be explored.

References

[1] L. Ehrlinger and W. Wöß, “Towards a definition of knowledge graphs.,” in *SEMANTiCS (Posters, Demos, SuCCESS)*, 2016.

[2] “Resource description framework (rdf).” https://www.w3.org/RDF/.

[3] “Sparql is a recommendation.” https://www.w3.org/blog/SW/2008/01/15/sparql_is_a_recommendation/.

[4] M. Yahya, K. Berberich, S. Elbassuoni, M. Ramanath, V. Tresp, and G. Weikum, “Natural language questions for the web of data,” in *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL ’12, (Stroudsburg, PA, USA), pp. 379–390, Association for Computational Linguistics, 2012.

[5] C. Liang, J. Berant, Q. Le, K. D. Forbus, and N. Lao, “Neural symbolic machines: Learning semantic parsers on freebase with weak supervision,” *arXiv preprint arXiv:1611.00020*, 2016.

[6] P. Trivedi, G. Maheshwari, M. Dubey, and J. Lehmann, “Lc-quad: A corpus for complex question answering over knowledge graphs,” in *International Semantic Web Conference*, pp. 210–218, Springer, 2017.

[7] “Question answering over linked data (qald).” http://qald.aksw.org/.

[8] T. Soru, E. Marx, A. Valdestilhas, D. Esteves, D. Moussallem, and G. Publio, “Neural machine translation for query construction and composition,” *arXiv preprint arXiv:1806.10478*, 2018.

[9] K. Papineni, S. Roukos, T. Ward, and W.-J. Zhu, “Bleu: a method for automatic evaluation of machine translation,” in *Proceedings of the 40th annual meeting on association for computational linguistics*, pp. 311–318, Association for Computational Linguistics, 2002.