Progress Report

Vinh Thinh Ho

December 17, 2018

Below, I would like to give a short summary about my research project Quantity Search (QSearch). First, I give a report about my current progress working on this project, and second, I define some goals to achieve in the next year.

1 Current Progress

The objective of the project is to answer quantity-related queries such as: *"cars with price less than $100k"*. There are also more complex queries that we want to answer, however I will not discuss here.

A such quantity-based search engine consists of the following building blocks:

- **Information sources**: define the source of information where we want to perform the search to answer the queries. Two common sources are web-tables and texts. We can also work with a mixture between these 2 sources.
- **Information extracting**: starting from raw information sources (web-tables and texts), this component extracts useful facts, standardizes, and stores them, which would be used for answering queries later on.
- **Query parsing**: people usually pose questions in form of free text, where the same question intent could be described in multiple ways, especially when the question is quantity-related. For instance, *"100.000 USD"* could be written as *"$100k"* or *"100k US dollars"*. Query parser canonicalizes the queries into the right form, which can be understood and hence answered.
- **Query answering**: from the information extracted from web-tables/texts and the standardized query, this component finds the relevant information for the queries, then filtering, aggregating and ranking to give users the final answers they are looking for.

1.1 Related Work

There exists many different related works tackling different aspects of the quantity search project. However, most of them work only with web-tables. In [5],

given a quantity-related query consisting of 2 parts: a_q is a verbal description of the response quantity type (e.g. **co2 emissions**); and e_q indicates an entity for which a quantity attribute is being sought (.e.g **china**). Then, the answer for this query is determined by looking up from the web-tables. [2] devises a probabilistic graphical model to canonicalize web-tables cells onto concepts, classes, entities and uniquely represented quantities in a knowledge base. [6] automates the task of *"entity augmentation"*, where it focuses on gathering numeric attributes about entities of interest by leveraging the vast corpus of web-tables. Working with free texts, the Open IE system BONIE [4] uses bootstrapping to learn patterns that express numerical relations in a sentence, and then uses them to deduce quantity-related facts.

1.2 Where Are The Spaces?

There are still a lot of works related to quantity that we can dig deeper. In [3], given a set of Web tables $T_1, ..., T_n$, and a query Q with q sets of keywords $Q_1, ..., Q_q$, decide for each T_i if it is relevant to Q and if so, identify the mapping between the columns of T_i and query columns. However, this work does not work on quantity, where we can spot on. There are also many other researchs working with web-tables, nevertheless not supporting quantity, which might open a path for us.

In most of existing works, a really good quantity extractor is still missing, especially for measures in domains such as medical, physics, chemistry, biology, etc. Focusing on this research direction might also produce promising result, even though the most difficult part is the sources of data, which is not usually published.

1.3 My Focused Work

In this section, I will described my focused work in details. My current work focuses on answering quantity-related queries , where the quantity is explicitly given (e.g. *"cars with price between 50k and 100k dollars in the USA"*; where the following is not supported now: *"building higher than eiffel tower"*).

For the query parsing, the idea is to parse the query into 3 fields:

- **Type T**
- **Quantity Q**: consists of a value **V**, an unit **U** and a comparison operator **O** (exact, approximate, lower/upper bound, range)
- **Context X**: a bag of words describing the connection between **T** and **Q**

The above question is parsed as follows:
- **T**:car; **Q**:(**V**:50k-100k; **U**:US dollar; **O**:range); **X**:{price, USA}

To answer this query, I focus on gathering information from texts. Similarly, we also extract information from texts and parse them into 3 fields:

- **Entity E**

- **Quantity Q**: consists of a value **V**, an unit **U** and a value resolution **R** (exact, approximate, lower/upper bound, range)
- **Context X**: a bag of words describing the connection between **E** and **Q**

For the sentence *"BMW i8 has price of $150k in the USA and range from 53 to 55km battery only"*, 2 facts are extracted:
- **E**:BMW i8; **Q**:(**V**:150k; **U**:US dollar; **R**:approximate); **X**:{price, USA}
- **E**:BMW i8; **Q**:(**V**:33-35; **U**:US dollar; **R**:range); **X**:{range, battery, only}

Now, the problem is at checking how to match between the query triples and the fact triples. In particular, we need to check if the entity **E** is an instance of the type **T**, if the fact quantity $\mathbf{Q}_{fact}$ satisfies the query quantity $\mathbf{Q}_{query}$ and if the fact context $\mathbf{X}_{fact}$ matches the query context $\mathbf{X}_{query}$.

1.3.1 Extraction from Texts

Extraction from texts is a crucial task in this proposal. To solve this problem, I propose to go through the following step:

- **Annotating quantity**: quantity mentions will be detected and replaced by a placeholder (e.g. *"<QUANTITY>"*). For example, the previous sentence is changed as:
  - *"BMW i8 has price of <QUANTITY> in the USA and range <QUANTITY> battery only"*
  For now, I reuse the Illinois Quantifier to annotate quantities from sentences.
- **Detecting entity**: entities from sentences are detected and linked to a KB, having a rich amount of type information, which we can leverage for searching part later on.
- **Tagging**: I propose to formalize the problem as a sequence tagging task and use a seq2seq model to solve. Figure 1 gives an overview about the Bi-LSTM model used to tag the processed sentence, which is simplified but also extended from [1]. This is a measure-specific tagging model, having 3 main layers: feature layer, Bi-LSTM layer, and softmax layer. Each feature vector of the input is the concatenation of 3 sub-feature vectors: *pre-trained word embedding*, *quantity position*, and the *entity feature*. Without the entity feature, this model emsembles many SRL models, where the position of the verb instead of the quantity is given. I would not go into detail implementation of the model.

  In the output, there are 4 tags being used: <*E*> for the entity connected to the quantity, <*X*> for the context tokens, <*QT*> for the quantity, and <*O*> for others. With entities spanning multiple tokens, only the first token is tagged with <*E*>, while the subsequent tokens are tagged with <*O*>.

  Constraints Decoding: In the predicting phase, each token is associated with a probability it belongs to each of the 4 tags. However, picking the tag with highest probability for every token does not generate a valid tag sequence, because we need to ensure the following 2 constraints:

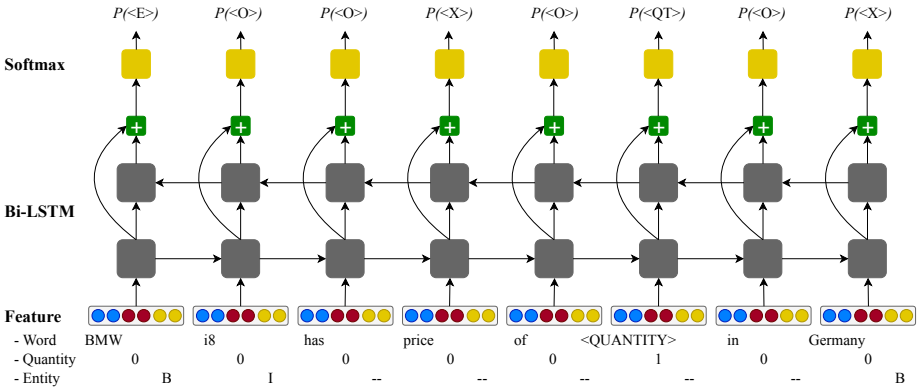

Figure 1: Overview of the Bi-LSTM quantity extraction model.

– There is at most 1 entity linked with the quantity. In the other word, there is at most 1 tag $<E>$, and it must be at the beginning of the linked entity (in case the entity spans over multiple tokens).

– There is exactly 1 tag $<QT>$ linked with the corresponding token at the quantity position of the input.

So, a decoding layer would be added in the top of the LSTM model to pick the sequence with the highest probability that satisfies the above constraints. This is easily done by using dynamic programming to build the senquence from the left to the right. We define:

$$F_{i,j,k,l} \; \forall \; i \in \{1..n\}; j \in \{<E>, <Q>, <X>, <O>\}; k, l \in \{0, 1\};$$

is the tag subsequence from position 1 to i with the highest probablity, where the token at position i is tagged with tag j, there currently exists k tags $<E>$ and l tags $<QT>$. Computation of $F_{i,j,k,l}$ could be done from previous states $F_{i-1,j',k',l'}$. The tag sequence we are looking for could be retrieved by tracing back from the last states $F_{n,j,k,1}$, where n is the length of the sequence.

2 Future Plans

2.1 Technical Plans

So far I have summarized what I have done. There are still many things to do here.

- First, the quantity extractor Illinois Quantifier I am using is not so good. The extraction has a lot of noise and does not support many measures. Since the quantity extraction is not my focus right now, however being a very crucial part for achieving a good result, I will restrict to work with

only some measures (such as length, money, percetage, etc.). This is also the reason why the model I am building is measure-specific. Even with these measures, the tool is still not perfect. So, in the short future, I plan to make a small tool quantity extractor. This tool should be very simple and does not consume much time. I am thinking about improve the output of Illinois Quantifier on the chosen measures.

- Second, another step is to annotate evaluation data for the learning model. This is time consuming, and a good number of test samples is still not clear for me now. For example, there might be about 200 samples for each measure, and if I work with 5 measures, then there will be total 1000 samples. The higher number of test samples, the better.

- Third, I have started looking a bit on the search part. Given a type description from query, returns all entites of that type from KB; and, given 2 pieces of contexts (query context and sentence context), check whether they match to each other. Obviously, the solution could be looked up from using word embedding, or Wordnet.

- Forth, collecting query for evaluation is also important. This is still unclear for me on how to do this. The queries should be realistic, measure-specific, and fits with our data sources.

- Finally, the current approach restricts ourselves to search for only entities in KB. However, it is great if we can go out of the KB, which is more challenging. This could be what I would do in the long future.

2.2 Non-Technical Plans

I would like to submit something next year. I am not sure if this plan is possible, since there are still many things to do, and I feel that the result depends strongly on the data we have. Good candidates for the conferences might be CIKM or WSDM.

References

[1] L. He, K. Lee, M. Lewis, and L. Zettlemoyer. Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 473–483, 2017.

[2] Y. Ibrahim, M. Riedewald, and G. Weikum. Making sense of entities and quantities in web tables. In *Proceedings of the 25th ACM International on Conference on Information and Knowledge Management*, pages 1703–1712. ACM, 2016.

[3] R. Pimplikar and S. Sarawagi. Answering table queries on the web using column keywords. *Proceedings of the VLDB Endowment*, 5(10):908–919, 2012.

[4] S. Saha, H. Pal, et al. Bootstrapping for numerical open ie. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, volume 2, pages 317–323, 2017.

[5] S. Sarawagi and S. Chakrabarti. Open-domain quantity queries on web tables: annotation, response, and consensus models. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 711–720. ACM, 2014.

[6] M. Zhang and K. Chakrabarti. Infogather+: semantic matching and annotation of numeric and time-varying attributes in web tables. In *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, pages 145–156. ACM, 2013.