Entities with Quantities: Extraction, Search, and Ranking

Vinh Thinh Ho
hvthinh@mpi-inf.mpg.de
Max Planck Institute for Informatics
Saarbrücken, Germany

Koninika Pal
kpal@mpi-inf.mpg.de
Max Planck Institute for Informatics
Saarbrücken, Germany

Niko Kleer
nkleer@mpi-inf.mpg.de
Max Planck Institute for Informatics
htw saar
Saarbrücken, Germany

Klaus Berberich
kberberi@mpi-inf.mpg.de
Max Planck Institute for Informatics
htw saar
Saarbrücken, Germany

Gerhard Weikum
weikum@mpi-inf.mpg.de
Max Planck Institute for Informatics
Saarbrücken, Germany

ABSTRACT

Quantities are more than numeric values. They represent measures for entities, expressed in numbers with associated units. Search queries often include quantities, such as athletes who ran 200m under 20 seconds or companies with quarterly revenue above $2 Billion. Processing such queries requires understanding the quantities, where capturing the surrounding context is an essential part of it. Although modern search engines or QA systems handle entity-centric queries well, they consider numbers and units as simple keywords, and therefore fail to understand the condition (less than, above, etc.), the unit of interest (seconds, dollar, etc.), and the context of the quantity (200m race, quarterly revenue, etc.) As a result, they cannot generate the correct candidate answers. In this work, we demonstrate a prototype QA system, called Qsearch, that can handle advanced queries with quantity constraints using the common cues present in both query and the text sources.

KEYWORDS

Semantic Search, Question Answering, Information Extraction, Quantities

ACM Reference Format:
Vinh Thinh Ho, Koninika Pal, Niko Kleer, Klaus Berberich, and Gerhard Weikum. 2020. Entities with Quantities: Extraction, Search, and Ranking. In *The Thirteenth ACM International Conference on Web Search and Data Mining (WSDM '20), February 3–7, 2020, Houston, TX, USA.* ACM, New York, NY, USA, 4 pages. https://doi.org/10.1145/3336191.3371860

1 INTRODUCTION

Motivation. Search engines have gone a long way towards understanding entities and their types in online contents and user queries, by learning information extraction and leveraging their knowledge graphs [14]. For example, a query about *"Bezos's net worth"* returns the direct answer 113.5 Bio. USD by recognizing the entity Jeff Bezos. Likewise, a query with a type description like *"athletes who won 200m olympics"* returns a crisp list of entities, including Usain Bolt and others.

These capabilities for semantic search [3, 4, 7] have limitations, though. One of them is the limited understanding of *quantities*, that is, quantitative measures of entities with a value and a unit, e.g., the personal wealth of a person, the annual revenue of a company, the time of a 200m race's winner, the fuel consumption of a car, and so on. Looking up quantities for given entities is straightforward and well supported. However, evaluating search conditions about quantities is beyond the abilities of today's search engines. Consider example queries such as:

- CEOs with a net worth of more than 100 Billion dollars
- Athletes who ran 200m under 20 seconds
- Hybrid cars with a battery range above 50 km

Depending on the values in these queries and the way the units are specified, major search engines occasionally return good results for some of the queries. For example, the third query, with the twist that the unit is changed to miles, returns a short list of car models ranked by electric range, however all these are from a list of a single web page. Changing the condition of the query from 50 km to 45 km returns a different list where all results are incorrect (their electric range is lower than 45 miles). If we change the value to 57, the search engine falls back to the ten-blue-links mode and returns web pages rather than entities. The underlying reason for these deficiencies is that the search engines do not understand quantities; instead they merely match tokens that denote numbers and units as if they were keywords. The occasional cases with good results are, by and large, accidental (drawing from many high-quality lists and tables within web pages).

Proposed Approach. The prototype system presented in this demo paper, called *Qsearch*, is an approach to overcome the outlined limitations. Qsearch supports queries about entity types with search conditions about quantities. The queries can take the form of keywords, telegraphic phrases or full-fledged questions. The above examples can be handled by Qsearch. Each produces ranked lists of entity-level results with evidence sentences that the quantity condition is satisfied. Measures within the scope of the prototype include financial quantities (e.g., company revenues or earnings or prices of company acquisitions), physical and technical measures of

WSDM '20, February 3–7, 2020, Houston, TX, USA

ACM ISBN 978-1-4503-6822-3/20/02…$15.00
https://doi.org/10.1145/3336191.3371860

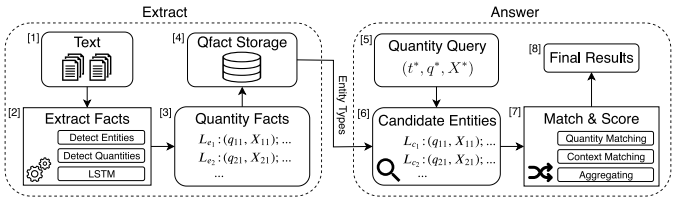

Figure 1: Overview of Qsearch [9].

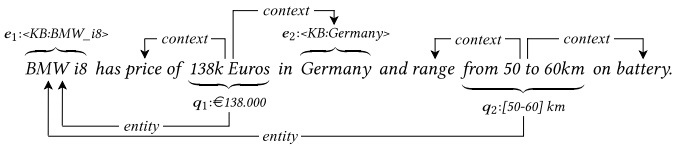

Figure 2: Each detected quantity is linked to a corresponding entity and relevant context tokens.

devices (e.g., range or fuel consumption of cars, battery capacity of mobiles, energy consumption of data centers), times and distances in sport competitions, and more.

Qsearch is based on three main components (see [9] for full details). First, text corpora are processed with a neural method for extracting quantities, the entities to which they refer, and informative cue words from their joint context, which are represented in the form of triples (e, q, X) - *(entity, quantity, context)*, called *Qfact*. For instance, from the text *"Tesla S costs 65k Euros in Germany"*, we extract Qfact: (*Tesla S*; *€65.000*; {*costs, Germany*}). Second, a query is decomposed into *entity type*, *quantity condition* and important *context* cue words, which are also casted into a triple (t^*, q^*, X^*), called *Qquery*. For example, query *"Cars with price less than €35,000 in Germany"* is modeled as Qquery: (*car*; *<€35.000*; {*price, Germany*}). The query processing matches Qquery against extracted Qfacts from text snippets. Third, the query results are judiciously ranked using language models or other criteria (e.g., entity prominence or quantity value).

Outline. In Section 2, we briefly describe the architecture of Qsearch and the approach underlying its main components. Afterwards, we illustrate the features provided by our demonstration through graphical user interface in Section 3. Finally, Section 4 mentions related work and Section 5 concludes the paper. We make our demo available to the research community at the following URL: https://qsearch.mpi-inf.mpg.de/

2 SYSTEM OVERVIEW

Figure 1 gives an overview of the architecture of Qsearch that has been introduced in our earlier work [9]. Qsearch consists of two main phases: *Extract* and *Answer*.

2.1 Extract

We preprocess to recognize named entities in the input text corpus (Block 1) using AIDA [11] system that links entity mentions to the YAGO knowledge base [21]. We also detect quantity mentions in the text and normalize them into standard units. For this, we make use of the Illinois Quantifier [17], a state-of-the-art tool for recognizing quantities in text, along with some hand-crafted rules (e.g., regular expressions).

Subsequently, we run a specifically designed Long Short Term Memory (LSTM) network to extract Qfacts in the form of triples (e, q, X) from each individual preprocessed sentence (Block 2). The neural network structure has been described in our work [9]. The main idea is that, for each quantity detected in the preprocessing step, the neural network identifies the entity to which it refers and the relevant context tokens that express the relation between them. Figure 2 illustrates an example of how Qsearch extracts Qfacts from sentences. In this example, the quantity q_1 is mapped to the entity e_1 and context {*price, Germany*}, whereas the quantity q_2 is also mapped to the entity e_1 but with context {*range, battery*}.

After the neural-based extraction, all Qfacts having the same named entity are grouped together, so that each individual entity e_i is connected with a list of quantities and contexts, expressed as $L_{e_i} = \{(q_{i1}, X_{i1}), (q_{i2}, X_{i2}), ...\}$ (Block 3). Entities are also linked to their semantic types from the KB, and we store these structures in a data repository in Block 4. In our implementation, Elasticsearch is used as the storage engine.

2.2 Answer

In this phase, Qqueries are answered by matching against Qfacts extracted from before.

Query Parsing. In Block 5, the input questions are transformed into Qquery format (t^*, q^*, X^*) by a rule-based parser. The parser uses a dictionary of YAGO types and a dictionary of quantity units to recognize the target semantic type of answers t^* and the quantity condition q^*. Then, it includes all remaining tokens of the query (except stopwords) in the query context X^* [9].

Answer Matching. First, based on the type information from the KB, a list of candidate entities $C = \{c_1, c_2, ...\}$ that satisfy the target type t^* are retrieved from the Qfact repository, along with their quantity-context pairs $\{L_{c_1}, L_{c_2}, ...\}$ (Block 6). Second, we perform quantity matching with the support of a list of unit conversion rules, filtering all Qfacts that do not meet the quantity condition q^* (Block 7). Finally, each candidate entity $c \in C$ is assigned a score calculated based on matching contexts X against X^*. Here, we make use of two context ranking models for measuring the context relevance: *Kullback-Leibler divergence (KL)* and *context embedding distance (ced)* (see [9] for details).

Kullback-Leibler Divergence. The Kullback-Leibler divergence between the query context X^* and the fact context X is defined as:

$$KL(X^*, X) = H(X^*, X) - H(X^*)$$
$$\equiv H(X^*, X) = -\sum_{w \in \mathcal{V}} P(w|X^*) \log P(w|X)$$

where $\mathcal{V}$ is the word vocabulary, $H(X^*)$ is the entropy of X^*; $H(X^*, X)$ is the cross entropy between X^* and X; and $\equiv$ is rank equivalence (i.e., preserving order, $H(X^*)$ is omitted). The two probabilities $P(w|X^*)$ and $P(w|X)$ are estimated using Maximum Likelihood Estimation (MLE) with WordNet-based context expansion and Jelinek-Mercer smoothing to avoid zero probabilities [9].

Context Embedding Distance. The novel context embedding distance for measuring context relevance is defined as follows:

$$ced(X^*, X) = ded(X^* \rightarrow X) \times ded(X \rightarrow X^*)^{\alpha}$$

Figure 3: Qsearch Web interface.

Figure 4: Type suggestion in Qsearch.

where $ded(A \rightarrow B)$ is the *directed embedding distance* between two bags of words A and B, computed as:

$$ded(A \rightarrow B) = 1 + \left(\sum_{u \in A} W(u) \min_{v \in B}(d(u,v)) \right) / \sum_{u \in A} W(u)$$

where $d(u,v)$ is the semantic distance between two words u and v, computed using word embeddings [15]; $W(u)$ is the importance weight of u; in this case, *inverse document frequency (idf)* is used by Qsearch. Putting into words, *ced* consists of two parts: (1) $ded(X^* \rightarrow X)$ measures how well query context tokens match with fact context, and (2) $ded(X \rightarrow X^*)$ captures how much other terms in X change its meaning, and thus, should be penalized in the total score. $\alpha \geq 0$ controls the influence of this penalty.

Entity Scoring. After applying entity-type and quantity filters, each entity is assigned a score from one of its linked Qfacts, which has the best context distance to the Qquery. Then, they are ranked by their scores to produce the final results (Block 8).

3 DEMONSTRATION

We developed a Web interface to allow users to experience our Qsearch system. Figure 3 shows a screenshot of the top search results for an example quantity query.

Input. Users can provide query in natural language text as input to the system. To give better search experience, we guide users by showing a list of top YAGO types along with the number of relevant entities existing in our database while writing the query. Figure 4 illustrates this feature.

Output. Qsearch processes the input query and generates a ranked list of entities relevant to the input query. The parsed query, top answers and their entity pages (from Wikipedia) are shown to users, along with the text snippets from where the Qfacts are extracted, and the links to the original web pages (see Figure 3). For better experience, we also show the representation images of the entity answers (extracted from the entities' Wikipedia pages) to the users when they hover on the entity links. For each answer, Qsearch highlights the extracted entity, quantity and context cues of the Qfact, and also provides the converted quantity value if its unit is different from the one being asked in the query. In Table 1, we provide some anecdotal examples of user queries and their top answers produced by Qsearch.

Table 1: Sample quantity queries and results from Qsearch.

Query		
Q1: Skyscrapers with height above 1000 feet		
Q2: Footballers with transfer cost more than 50 million Euros		
Q3: Sprinters who ran 100 meter in less than 10 seconds		
Query	Result	Corresponding Sentence
Q1	Empire State Building	102nd Floor Building: Empire State Building, New York Height: 1,250 feet.
	Mercury City Tower	Eleven high-rises have already been built, among them the golden Mercury City Tower, at a height of 339 meters, the tallest skyscraper in Europe.
Q2	Neymar	The judge says Neymar cost at least 83.3 million euros ($88 million), while Barcelona insists it paid 57 million euros (then $74 million).
	Raheem Sterling	Raheem Sterling became the most expensive English footballer ever on Tuesday after joining Manchester City from Premier League rival Liverpool in a drawn-out move costing 49 million pounds ($114 million).
Q3	Andre De Grasse	Andre De Grasse, a 20-year-old from Markham, Ont., has run the 100 metre in under 10 seconds three times this year.
	Walter Dix	Walter Dix, a Florida State sprinter, ran the 100 meter dash in 9.93 seconds, the fastest time in the world this year and the second fastest ever by a collegian.

Exploration of underlying data sources. Qsearch answers quantity queries based on information from three large collections of

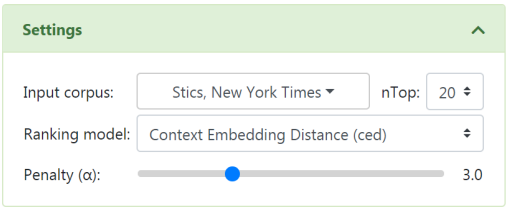

Figure 5: Qsearch options.

text documents - two real-world news corpora, the *STICS* project [10] (containing 5.84M documents) and the *New York Times* archive [19] (containing 1.77M documents), and a collection of web pages from the English Wikipedia[1] (containing 5.78M documents). In total, our text data consists of 13.39M documents. By default, Qsearch uses all three collections to generate answers. However, we allow users to change this setting (see Figure 5) in order to narrow down their search on specific text collections. As the characteristic of news articles is quite different from Wikipedia articles, using both types of text collections as underlying input data allows Qsearch to answer a wide range of quantity queries. For example, information about finance or sport domain can be found all over the news articles and also in Wikipedia, but answers for queries on geological objects (e.g., glaciers with length more than 100 km) are mainly covered only by Wikipedia articles.

Exploration of answer generation methods. Qsearch employs different ranking models as mentioned in Section 2 to rank the entity answers and shows top confident results to the users. In the default configuration, Qsearch uses the *context embedding distance (ced)* as the ranking model, with the penalty factor $\alpha = 3$ (as in [9]), and shows top 20 answers. We provide a flexible interaction to the system by allowing users to modify these search settings (as shown in Figure 5) to explore the answer generation methods. Here, users can adjust the number of retrieved results (10, 20, 30, 40 or 50) or change the underlying ranking model (*context embedding distance* or *KL-divergence*) and its parameter. Additionally, Qsearch provides further exploration with a secondary ranking criterion, where top retrieved answers can be re-ranked either by quantity value (after normalization), or by entity prominence (based on view count of entities' Wiki pages). This secondary ranking criterion can be set using *"Sort by"* button in Figure 3.

4 RELATED WORK

Prior work has looked into recognizing and extracting numeric expressions from text using techniques like CRFs and LSTMs (e.g., [1, 17, 18]). However, this alone does not turn numbers into interpretable quantities, as the latter require units (e.g., MPG, kWh/100km, etc.) and proper frame of reference (e.g., the timeframe for revenue, dosages etc.). Only few works attempted to canonicalize quantities by mappings to hand-crafted knowledge bases of measures [13], but these efforts are very limited in scope. The most promising one has been the work by Sarawagi et al. on querying Web tables [20], with limited support for understanding measures and units. However, this approach is tied to table data and does not generalize to quantity search over all kinds of Web contents.

[1] https://meta.wikimedia.org/wiki/Data_dump_torrents#English_Wikipedia

Semantic role labeling is a slot-filling variant of information extraction, intensively studied in NLP. Modern methods leverage word embeddings and deep learning [8]. Our extraction method uses the similar paradigm, but customizes the learning to our specific model of Qfacts.

Semantic search (e.g., [2–4, 7]) has largely focused on entity-seeking queries, where either a type description or an entity name is the starting point. When such requests involve quantities, it is in the form of simple lookups, for example, the net worth of Bezos or the range of the Tesla Model S. These queries do not require any understanding of measures and search conditions over quantities. The same holds for most of the work on question answering, where natural language questions are mapped into queries over knowledge graphs or text corpora (e.g., [5, 6, 12, 16]). Although some benchmarks like QALD and ComplexWebQuestions include a small fraction of questions that mention quantities, this is almost exclusively in the form of lookups (e.g., what is Bezos's net worth) rather than search conditions (e.g., CEOs with net worth above 1B).

5 CONCLUSION

We have presented the Qsearch demo system for full-fledged support of quantity queries, with novel approaches of information representation, information extraction and answer matching/ranking. We capture quantities in complete extent, including measurement units, reference entities and appropriate contexts. In the future, we plan to enhance the work further by integrating more data sources and provide better Web interface to the users.

REFERENCES

[1] O. Alonso and T. Sellam. Quantitative information extraction from social data. SIGIR 2018.
[2] K. Balog. *Entity-Oriented Search. The Information Retrieval Series*, Springer, 2018.
[3] H. Bast, B. Buchhold, and E. Haussmann. Semantic search on text and knowledge bases. *Foundations and Trends in Information Retrieval*, 2016.
[4] H. Bast and N. Schnelle. Efficient and convenient sparql+text search: A quick survey. Reasoning Web 2018.
[5] C. Clark and M. Gardner. Simple and effective multi-paragraph reading comprehension. ACL 2018.
[6] D. Diefenbach et al. Core techniques of question answering systems over knowledge bases: a survey. *Knowl. Inf. Syst.*, 2018.
[7] D. Garigliotti and K. Balog. Towards an understanding of entity-oriented search intents. ECIR 2018.
[8] L. He et al. Deep semantic role labeling: What works and what's next. ACL 2017.
[9] V. T. Ho, Y. Ibrahim, K. Pal, K. Berberich, and G. Weikum. Qsearch: Answering quantity queries from text. ISWC 2019.
[10] J. Hoffart, D. Milchevski, and G. Weikum. STICS: searching with strings, things, and cats. SIGIR 2014.
[11] J. Hoffart et al. Robust disambiguation of named entities in text. EMNLP 2011.
[12] X. Huang et al. Knowledge graph embedding based question answering. WSDM 2019.
[13] Y. Ibrahim, M. Riedewald, and G. Weikum. Making sense of entities and quantities in web tables. CIKM 2016.
[14] N. F. Noy et al. Industry-scale knowledge graphs: lessons and challenges. *Commun. ACM*, 2019.
[15] J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. EMNLP 2014.
[16] P. Rajpurkar et al. Squad: 100, 000+ questions for machine comprehension of text. EMNLP 2016.
[17] S. Roy, T. Vieira, and D. Roth. Reasoning about quantities in natural language. *TACL* 2015.
[18] S. Saha, H. Pal, and Mausam. Bootstrapping for numerical open IE. ACL 2017.
[19] E. Sandhaus. The new york times annotated corpus, 2008.
[20] S. Sarawagi and S. Chakrabarti. Open-domain quantity queries on web tables: annotation, response, and consensus models. KDD 2014.
[21] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. WWW 2007.