Extracting Contextualized Quantity Facts from Web Tables

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

Simon Razniewski
srazniew@mpi-inf.mpg.de
Max Planck Institute for Informatics
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

Quantity queries, with filter conditions on quantitative measures of entities, are beyond the functionality of search engines and QA assistants. To enable such queries over web contents, this paper develops a novel method for automatically extracting quantity facts from ad-hoc web tables. This involves recognizing quantities, with normalized values and units, aligning them with the proper entities, and contextualizing these pairs with informative cues to match sophisticated queries with modifiers. Our method includes a new approach to aligning quantity columns to entity columns. Prior works assumed a single subject-column per table, whereas our approach is geared for complex tables and leverages external corpora as evidence. For contextualization, we identify informative cues from text and structural markup that surrounds a table. For query-time fact ranking, we devise a new scoring technique that exploits both context similarity and inter-fact consistency. Comparisons of our building blocks against state-of-the-art baselines and extrinsic experiments with two query benchmarks demonstrate the benefits of our method.

KEYWORDS

Information Extraction, Quantity Facts, Web Tables

ACM Reference Format:
Vinh Thinh Ho, Koninika Pal, Simon Razniewski, Klaus Berberich, and Gerhard Weikum. 2021. Extracting Contextualized Quantity Facts from Web Tables. In *Proceedings of the Web Conference 2021 (WWW '21), April 19–23, 2021, Ljubljana, Slovenia.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3442381.3450072

1 INTRODUCTION

Motivation. A good fraction of web queries revolve around quantities of entities: looking up, filtering, comparing and aggregating quantitative properties such as heights of buildings, running times of athletes, goals or scoring rates of footballers, energy consumption of electric cars, etc. [4, 7, 16]. In this paper we focus on *quantity filters* [16, 17], an important class of queries and also a building block for comparative search. Examples are:

- British football teams worth more than 1.5 billion pounds
- sprinters who ran 100 m under 9.9 seconds
- electric cars with energy efficiency above 80 MPG-e

Note that this kind of query is more difficult than quantity lookups, such as "the value of Manchester City" or "the personal 100 m record of Usain Bolt". Lookups are well supported by search engines and QA assistants. Quantity filters, on the other hand, lack this support as conditions like "more than 1.5 billion pounds" or "under 9.9 seconds" are mostly interpreted in string-matching mode. For some examples, search engines return good web pages, such as Wikipedia articles on "10-second barrier" or "100 metres", but this is not the user's query intent and she has to tediously sift through these pages rather than receiving a crisp entity-list answer. Moreover, the result quality depends on the value in the query, as some (string-interpreted) values match good list pages. For example, there is a list of 100m races under 10 seconds, but none ready for 9.9, 9.8, etc.

Table 1: Illustrative example on football teams.

Team	Stadium	Capacity	Coach	Value (in Bio)
Bayern	Allianz Arena	ca. 75000	Hansi Flick	2.549 Euro
Real	Bernabéu	81,044	Zidane	3.649 Euro
Man City	unknown	n/a	Pep Guardiola	2.055 GBP
Chelsea	Stamford Bridge	40,834	Frank Lampard	1.958 GBP
Liverpool	Anfield	53,394	Jürgen Klopp	ca. 1.7 GBP

Instead of tapping the web, we could turn to knowledge bases (KBs) and structured sources in the Open Data ecosystem. However, KBs hardly cover quantities; for example, Wikidata contains thousands of sprinters but knows their personal records only for a few instances. To tap Open Data sources, one would still have to find the relevant datasets in a sea of data sources, and assess their freshness and completeness.

Problem. At the core of answering quantity-filter queries is the problem of extracting entity-quantity facts from web sources. This has been successfully addressed in [16] for the case of single sentences from text sources, by recognizing entity-quantity pairs along with relevant context words and building on prior work for spotting quantities with numeric values and units [31–33]. In this paper, we aim to tap into a different kind of data sources, namely, ad-hoc *web tables* embedded in HTML pages, and address the problem of accurately extracting entity-quantity facts with relevant context.

WWW '21, April 19–23, 2021, Ljubljana, Slovenia

ACM ISBN 978-1-4503-8312-7/21/04.
https://doi.org/10.1145/3442381.3450072

An illustrative example, which could serve to answer the query about British football teams, is shown in Table 1.

There are good prior works on extracting entity-centric facts from web tables, including surveys [6, 11, 41]. The output is typically a set of subject-predicate-object (SPO) triples, obtained by judiciously picking two cells in the same row as S and O and deriving P from the column header of O. In conjunction with entity linking to a KG [34], an extractor could yield, for instance, (*Real Madrid, hasCoach, Zinedine Zidane*).

However, state-of-the-art methods do not work well for quantity facts for several reasons:

- First, quantities appear in very diverse and potentially noisy forms. For example, the team values in Table 1 are just strings, varying in units and scale and missing values ("unknown", "n/a"). Proper interpretation of table cells may require understanding the surrounding text.
- Second, it can be hard to infer which column pair denotes a quantity fact, that is, to which entity column a quantity column refers. In the example Table 1, we need to determine that Capacity refers to Stadium and Value to Team, but this is not obvious for a machine. This is further aggravated by the common situation that column headers are more generic and less informative. For example, instead of headers like Team, Stadium, Capacity, etc., we could have Name, Site, Size, etc., which are hard to interpret. Prior works on web tables seemed to assume that all columns (for possible choices of O) refer to the same column (for S), and that this per-row-entity column is usually the leftmost one [6, 41]. However, these assumptions are not always true.
- Third, extracting entity-quantity pairs alone is not sufficient for query answering, as many queries include additional modifiers such as "British" or cues for the measure of interest such as "energy efficiency". To be able to match these against a repository of quantity facts, the fact extraction needs to capture also relevant context. Prior works on triples from web tables ignored this important issue; they viewed the extraction as uncoupled from downstream use cases like user queries and questions.

Approach. This paper addresses the outlined problems and presents a full-fledged solution, called **QuTE** (**Qu**antity **T**able **E**xtraction), for extracting contextualized quantity facts from web tables, to support quantity-filter queries. First, to cope with noisy quantities and diverse units and scales in tables, we employ pattern-based extractors and rule-based normalization. Second, for the problem of aligning the right pair of entity and quantity columns, one of the key tasks, we devise a statistical inference method that leverages external text corpora. Third, to contextualize the extracted quantity facts, we exploit text and DOM-tree markup that surround a table, and we introduce a novel way of computing confidence scores for quantity facts, based on evidence in text collections. Finally, as the resulting facts may still yield many false positives in query results, we have developed additional methods for enhanced scoring at query time based on consistency learning [39].

Contribution. The following are novel contributions:

- We present a robust solution for the column alignment problem posed by complex tables, by harnessing external text corpora and joint inference with entity linking. This is the first method specifically geared for extracting quantity facts, with the novel technique of leveraging cues from a large text corpus (Sections 3 and 4).
- We introduce a new way of computing quantity fact confidence scores, by incorporating evidence from text collection, with type-based inference to overcome sparseness problems (Section 4).
- We present a new method for corroborating extracted facts at query time, re-ranking them and pruning false positives based on a technique for consistency learning (Section 5).
- Experiments include comparative evaluations of our major building blocks against various baselines, and an extrinsic study of how well the extracted facts support quantity queries. The latter is based on a benchmark of 100 queries from [16] and a new collection of 150 queries with list-based ground-truth.

Experimental data and code are available at: https://www.mpi-inf.mpg.de/research/quantity-search/quantity-table-extraction.
A QuTE-based search demonstrator is accessible at: https://qsearch.mpi-inf.mpg.de/table/.

2 MODEL AND SYSTEM OVERVIEW

2.1 Model

The *input* for fact extraction is a collection of ad-hoc tables, from a web crawl, spreadsheet corpus or Wikipedia dump (e.g., [13]).

Definition [Web Table]. A web table with r rows and c columns is a tuple $T = (H, B, \mathcal{X})$ where:

- $H = \{h_i | i \in \{1..c\}\}$ are the headers of the c columns;
- $B = \{b_{i,j} | i \in \{1..r\}, j \in \{1..c\}\}$ are cells in the table body;
- $\mathcal{X}$ is the context surrounding the table, which typically includes web page title, table caption, DOM-tree headings for the HTML path to the table, and text in proximity to the table.

We denote $C_k = \{h_k\} \cup \{b_{i,k} | i \in \{1..r\}\}$ and $R_k = \{b_{k,j} | j \in \{1..c\}\}$ as the k-th column and k-th row, respectively.

This definition is geared for "horizontal" tables with column headers and row-wise records. For "vertical" tables with row headers and data records per column, we can detect the orientation and apply a transpose operation, using heuristics from [6].

Definition [E-column and Q-column]. For a given table, all columns whose cells predominantly contain named entities (which could be linked to a knowledge base) are referred to as E-columns. All columns whose cells predominantly contain numeric quantities are denoted as Q-columns. The implementation of "predominantly" is based on thresholds (say 80%) for the fraction of cells that qualify one way or the other. Columns that are neither labeled E nor Q (e.g., with many cells containing long text) are disregarded. In Table 1, the columns Team, Stadium and Coach are E-columns, and Capacity and Value are Q-columns.

The *output* of extracting facts from a table is represented in the form of triples called quantity facts, or **Qfacts** for short (cf. [16] where this terminology is defined for text-based extraction).

Definition [Qfact]. A quantity fact extracted from table $T = (H, B, \mathcal{X})$ is a triple of the form $\mathcal{F} = (e, q, X)$ where:

- e is an entity in a table-body cell $b_{i,j}$ of an E-column C_j, either in the string form of an entity mention or already in the form of a linked entity uniquely identified in a KB;

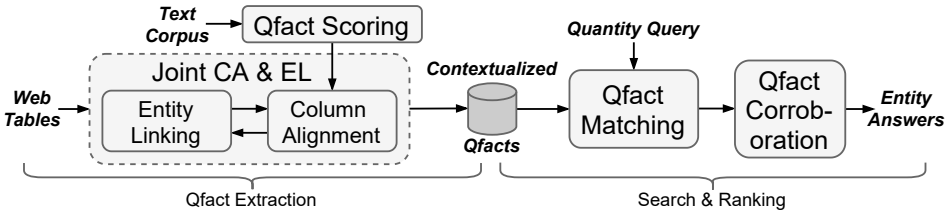

Figure 1: Overview of the QuTE system.

- q is a quantity, properly normalized and with proper unit, in a cell $b_{i,k}$ of a Q-column C_k;
- X is Qfact context, a (small) set of cue words (or phrases) extracted from the table (incl. context $\mathcal{X}$) that are specifically informative for the pair (e, q).

As an example, a perfect extractor from Table 1 should produce Qfacts such as (*Estadio Santiago Bernabéu, 81044, "stadium, capacity, seats, Madrid"*), (*Chelsea F.C., 1,958,000,000 GBP, "team, value, football, London"*), assuming informative text surrounding the table.

For the downstream use case of query answering, we consider a simple model of telegraphic or question-style queries containing a single quantity filter, following [16]:

Definition [Qquery]. A quantity query is a triple $\mathcal{Q} = (qt, qq, qX)$ where:

- qt is the expected type of answer entities, such as `football team` or `sprinter`;
- qq is a quantity condition of the form "θ *value unit*" where θ can be $\geq$, $\leq$, between, or (approximately) equal, and the unit is optional, as some measures do not have units, such as stadium capacity or country population;
- qX is a set of additional qualifier terms that an answer should match, such as "British" or "100 meters" or "Olympics", etc.

The *answer* to a Qquery is a Qfact that matches all query conditions, where context terms can be matched approximately (e.g., partially or by embedding-based similarity):

Definition [Qanswer]. An answer to a Qquery $\mathcal{Q} = (qt, qq, qX)$ is a Qfact $\mathcal{F} = (e, q, X)$ such that e is an entity of type qt, q satisfies the filter condition qq, and X is a sufficient match to the query context qX.

For example, the Qfact (*Chelsea F.C., 1,958,000,000 GBP, "team, value, football, London"*) would approximately match a query about "British football teams with value above 1.5 billion pounds" (as "British" and "London" are highly related by word embeddings).

2.2 System

All components of QuTE, i.e, Qfact extraction method along with a quantity-query processor and result ranker, are implemented in a pipeline depicted in Figure 1.

The pipeline starts with quantity recognition and normalization for Q-columns and entity linking to a KB for E-columns. A crucial step then is column alignment that links a Q-column with its proper E-column, to obtain a valid Qfact. Contextualization and scoring of Qfacts involves analyzing the context around a table and statistics from external corpora. Finally, query processing involves matching and an additional scoring step, taking inter-fact consistency into account.

For **quantity recognition**, we employ a combination of the prior works on QEWT [33] and Illinois Quantifier [31]. The latter is used to extract numeric values and units from table cells. QEWT is applied to the column headers to discover additional information about units and, possibly, scaling factors. Then, detected quantities are linked to the QuTree catalog [33] for normalization, including unit conversions.

For **entity recognition**, we employ the AIDA dictionary (*github.com/ambiverse-nlu*), which provides a large set of entity names, such as "Real", "Bayern", etc., and candidate entities.

For **entity linking (EL)** (i.e., disambiguating the recognized mentions onto KB items), there are ample prior works specifically geared for web tables [3, 14, 19, 23, 29]. We follow [3], with inference over a probabilistic graphical model. This takes into account a prior for entity popularity, context similarity between mentions in table cells and the KB entities, and the coherence among entity candidates for the same row (which should be semantically related entities) and the same column (which should be of the same semantic type). We denote result entities by Φ, with $\Phi(b_{i,j})$ is the entity for input mention $b_{i,j}$ in the table body.

3 COLUMN ALIGNMENT

A major building block of QuTE is the column alignment, which aligns a Q-column with its proper E-column, in order to extract Qfacts from the right pair of columns. This section discusses the limitation of prior works on web table processing and proposes a robust method for this task. Key novelties of our method are to leverage cues from an external text corpus, and to couple the inference for column alignments with the entity linker.

Definition [Column Alignment (CA)].
Given a pre-processed table T with x Q-columns $\{C_{k_1}, C_{k_2}, ..., C_{k_x}\}$ and y E-columns $\{C_{v_1}, C_{v_2}, ..., C_{v_y}\}$, a column alignment is a function Λ that maps each Q-column to one E-column:
$\Lambda = \{C_{k_i} \rightarrow C_{v_j} | \, i \in \{1..x\}\}$

3.1 Heuristics and their Limitations

Column alignment has been addressed in prior works [5, 8, 37] under simplifying assumptions, like mapping all Q-columns to the same E-column, which boils down to identifying a single subject column for the entire table. We overcome this limitation, but nevertheless consider heuristics that are inspired from prior works.

Definition [Leftmost Heuristic]. Each Q-column C_k is mapped to the leftmost E-column C_v, that is, the smallest v for which C_v qualifies as an E-column.

Definition [Closest-Left Heuristic]. Each Q-column C_k is mapped to the closest E-column C_v that is left of C_k, that is, $v < k$ and $k - v$ is minimal.

Definition [Most-Unique Heuristic]. Each Q-column is mapped to the E-column with the largest number of unique values (resembling a relational key). In case of a tie, pick the leftmost one.

In many cases, these three heuristics perform remarkably well, notwithstanding their simplicity. For our example in Table 1, they would be far from perfect, though. The Leftmost heuristic maps all Q-columns to Team. The Closest-Left heuristic correctly aligns Capacity to Stadium, but erroneously aligns Value to Coach. The Most-Unique heuristic does not help in this example, as all table cells have unique values.

Methods that consider multiple subject columns within the same table mostly rely on linking column headers to classes or concepts in a comprehensive knowledge base (e.g., [3, 14, 19, 23, 29]), to map column pairs to KB relations. However, quantity measures are covered only very sparsely in state-of-the-art KBs. As our goal is to cover a wide variety of quantity types, we cannot rely on the KB for column alignment. The only prior work for handling multiple subject columns and aligning other columns without assuming a prior KB is [5]. This method is based on discovering functional dependencies by analyzing entropy measures between columns. However, in Q-columns the typical situation is that all values are distinct, so that their frequencies are trivial and do not give hints for cross-column scoring. Moreover, we found that even when a web table has multiple E-columns, the values in all of them are often unique – as tables often have only few rows. Table 1 is a typical case, and the method of [5] does not add any benefit over the simpler heuristics here. Hence we disregard this method.

3.2 QuTE Method for Column Alignment

We propose a robust column alignment approach by modelling the connections between a pair of E-column and Q-column as a graph. To compute a *CA-score* for a candidate alignment, we devise a graph-based connectivity measure that considers the co-occurrence signals for same-row entity/quantity pairs, with entities chosen by the initial entity linking Φ. Essentially, we treat these entity/quantity pairs as Qfacts and leverage external corpus evidence to assess their confidence.

Definition [CA-score]. The quality of a column alignment Λ is:

$$\textit{CA-score}(\Lambda|\Phi) = \frac{1}{Z} \sum_{(C_k \to C_v) \in \Lambda} \sum_{\substack{(e,q) \text{ with} \\ e=\Phi(b_{i,v}),\ q=b_{i,k} \\ i=1..r}} \textit{ext-score}\big(\mathcal{F} = (e, q, X = h_k)\big)$$

where Z is a normalization constant and *ext-score*($\mathcal{F}$) is a score for observing a Qfact that e has the quantitative property $X{:}q$ in an external data collection, and X is the header of Q-column C_k.

Prior works on extracting SPO triples from web tables often resorted to pre-existing triples in a knowledge base as "witnesses" for the scoring of newly extracted facts (in the spirit of distant supervision). For our task, this idea would boil down to a chicken-and-egg problem, as we do not yet have a richly populated KB of quantities. Therefore, we harness a different source of external evidence, namely, large text corpora that potentially contain sentences about e having property $X{:}q$. Observations of this sort, with potential relaxation of the exact value q, are the basis for the computation of *ext-score*($\mathcal{F}$). We describe this building block in Section 4.

3.3 Iterative Learning of Column Alignment

Column alignment (CA) can be integrated with entity linking (EL) for joint inference. The rationale for tackling CA and EL jointly is that either one can give informative cues to the other, to arrive at a better solution. CA can build on the output of EL, by incorporating more precise information about the entities in a candidate E-column. To this end, it can test if the entities exhibit high relatedness with the header of the Q-column under consideration. For example, "Capacity" is rarely seen in combination with *Real Madrid*, *FC Bayern Munich*, etc., but it is often co-occurring with *Estadio Santiago Bernabéu*, *Allianz Arena*, etc. Conversely, if we already have a good CA solution, this can benefit the EL task by identifying more focused context. In particular, rather than considering all cells in the same row of an entity as equally relevant for per-row coherence, we could give higher weight to the coherence between cells of the aligned E-column and Q-column. For example, frequent co-occurrence of "Bernabéu" and the aligned cell "Capacity: 81,044" (in a text corpus, e.g., Wikipedia, possibly with 81,044 relaxed into any number around 80,000), could boost the linking to *Estadio Santiago Bernabéu* rather than the footballer and club president *Santiago Bernabéu* (after whom the stadium is named).

We incorporate these mutual benefits by devising a joint objective function as follows.

Definition [Plausibility Maximization]. We define the plausibility of interpreting table T with entity linking Φ and column alignment Λ as:

$$\lambda \cdot \textit{CA-score}(\Lambda|\Phi) \ + \ (1-\lambda) \cdot \textit{EL-score}(\Phi|\Lambda) \qquad (1)$$

where λ is a tunable hyper-parameter. Here, *EL-score*($\Phi|\Lambda$) is the collective inference of entity linking module considering E-column/Q-column pairs selected by Λ.

Inference Algorithm. For joint inference about CA and EL, we adopt the collective classification method from [24], called *ICA*, which was also used by [3]. This avoids the high complexity of full-fledged MRF inference, which would be prohibitive as our factor graphs are very dense.

In essence, for each column alignment Λ, we compute the best EL solution Φ conditioned on Λ using the ICA method. The pair (Λ, Φ) that maximizes the joint objective function (plausibility maximization in Equation 1) is chosen as the final result.

3.4 Contextualization of Qfacts

We extract Qfacts based on the optimal pair (Λ, Φ) computed from the joint inference model. All extracted Qfacts are contextualized with the Q-column header, informative cue words from table caption, same-row cells, page title, all DOM-tree headings leading to the table, and the text in proximity to the table (e.g., preceding and following paragraph). All these components are optional. This way, we capture cues such as "football clubs" for Table 1. We include all words from these context items, forming a bag-of-words. The final output is a Qfact in the form (e, q, X) with entity e, quantity q and *contextualization* X.

4 QFACT CONFIDENCE SCORING

This section explains how we utilize external text corpora to compute *ext-score*($\mathcal{F}$) for the CA-score model of Section 3.2.

The key idea is to retrieve evidence for a candidate Qfact $(e, q, X = h_k)$, spotted from a table with Q-column C_k, in a larger corpus of text, such as sentences from Wikipedia articles. To this end, we employ the text-based extraction method from [16]: a trained LSTM network classifies sentences that contain at least one entity and one quantity and tags proper pairs of entity and quantity, along with informative context words from the sentence. Running this on Wikipedia full-text, followed by removing duplicates and thresholding on confidence, we obtained a collection $\mathbb{C}$ of 1.6M million Qfacts triples in the form (e', q', X'). By using an entity coreference resolution tool (*github.com/huggingface/neuralcoref*) on two consecutive sentences and combining them into one input, we enlarged this to a total of 2.4M million Qfacts – with a fair amount of uncertainty, though.

We treat this collection $\mathbb{C}$ as external evidence against which we can assess Qfact candidates distilled from web tables. A candidate table-Qfact (e, q, X) is highly-confident if related information can be found in text, in particular, in $\mathbb{C}$.

Definition [Evidence Score]. For Qfact $\mathcal{F} = (e, q, X = h_k)$, the evidence score from collection $\mathbb{C}$ is:

$$\textit{ext-score}(\mathcal{F}) = \max_{(e', q', X') \in \mathbb{C} \wedge e' = e} sim\big((q, X), (q', X')\big)$$

$$\text{where } sim\big((q, X), (q', X')\big) = w_1 \cdot sim_1(q, q') + w_2 \cdot sim_2(X, X')$$

with tunable coefficients w_1, w_2. The function finds the best matching evidence Qfact in $\mathbb{C}$ with same entity e. sim_1 compares quantities q and q' (after normalization to the same unit) and returns a score that is equal to their relative numeric distance $\frac{|q-q'|}{\max(|q|,|q'|)}$. We consider a 1% difference as a perfect match, because quantity values are often rounded or truncated. Note that if the two quantities are incomparable (from different concepts, e.g., length vs. monetary), we do not consider (e', q', X') at all. sim_2 compares the Q-column header $X = h_k$ with the evidence context bag-of-words X' by the *directed embedding distance* of [16]. This rewards if the column name appears in X', but also gives credit to different words that are related by their word2vec embeddings.

Type-based Evidence. Many of the candidate facts from tables may not find any text-based evidence by the above procedure. This is natural, as we expect to obtain a large number of facts from tables that cannot be spotted in text corpora at all. If this were not the case, we would not need to tap into tables and could instead extract from text only.

We can relax our notion of text evidence, however, and settle for the softer task of spotting some Qfact evidence with the same entity *type* as the candidate at hand. For example, to scrutinize the candidate (*Estadio Santiago Bernabéu, 81044, "Capacity"*), we can consider the text evidence (*Old Trafford, 74140, "Capacity"*) or (*Camp Nou , 99354, "Capacity"*). Intuitively, for an entity of the same type `stadium`, the cue word "Capacity" is important and the respective quantities fall into the same order of magnitude. In contrast, when examining candidates (*Santiago Bernabéu, 81044, "Capacity"*) and (*Real Madrid, 81044, "Capacity"*), there is hardly any text evidence that a `person` or a `team` has a Capacity. This reinforces the hypothesis that the table-based candidate is valid, including the chosen EL target ("Bernabéu" refers to *Estadio Santiago Bernabéu*, not to the club president *Santiago Bernabéu*) and the CA inference (Capacity refers to Stadium, not to Team).

Definition [Type-based Evidence Score]. For candidate Qfact $\mathcal{F} = (e, q, X)$ and each entity e^* sharing the same type with e, we compute a type-based evidence score of $\mathcal{F}$ with respect to e^* as:

$$\textit{t-ext-score}(\mathcal{F}|e^*) = \max_{(e', q', X') \in \mathbb{C} \wedge e' = e^*} \textit{rel}(e, e^*) \cdot sim\big((q, X), (q', X')\big)$$

where $\textit{rel}(e, e^*)$ is the semantic relatedness between the two entities (*ext-score* is actually a special case when $\textit{rel} = 1$). The *rel* function can be based on distance measures in the underlying type taxonomy, or alternatively by the cosine between word2vec (or wikipedia2vec) embeddings. In our implementation, we chose the shortest Wu-Palmer taxonomy distance [38] between the direct types of two entities. This has the advantage that we can incorporate entities e^* incrementally in ascending order of distance. This way, we efficiently prune the huge space of potential evidence items.

Combining Scores. A good fraction of the table-based Qfact candidates may have both kinds of text evidence: matching entities and merely matching types. Thus, it is natural to combine both scores. We define the final evidence score of $\mathcal{F}$ as the average of matching evidence scores from top-k best entities e^* (including e itself). We hypothesize that this yields a more robust signal from the wealth of text-based evidence.

Table 2 shows examples of top-scoring text evidence for examining the Qfact candidate (*Allianz Arena, 75000, "Capacity"*).

5 USE CASE: QUANTITY QUERYING

5.1 Matching and Ranking

All Qfacts from web tables are fully contextualized into the form $\mathcal{F} = (e, q, X)$, stored and indexed. We process a Qquery $Q = (qt, qq, qX)$ against this data by mostly following the method of [16]: Qfact entities are matched against the target type qt using type information from the KB, quantities are compared to query condition qq, and the context agreement between X and qX is quantified by the *directed embedding distance* of [16]. This yields a ranking of entity answers to a given query.

We extend the context comparison, as our setting differs from [16]. In text-based Qfacts, the context tokens come from the same sentence or short snippet. In contrast, for the table-based setting, we combine a set of cues from different kinds of context: Q-column header, page title, table caption, DOM-tree headings, same-row cells, and surrounding text window. To reflect this heterogeneity, we assign tunable weights to the context tokens based on their origin.

Definition [Weighted Directed Embedding Distance].

$$w\textit{-ded}(X, qX) = \left(\sum_{u \in qX} \omega(u) \cdot \min_{v \in X} (\sigma(v) \cdot d(u, v)) \right) / \left(\sum_{u \in qX} \omega(u) \right)$$

with ω denoting tf-idf-based weights of tokens and $\sigma(v)$ denoting the weight of Qfact context token v depending on the kind of context from where it originates. We have six different σ weights for the six kinds of context considered (see above); they are not word-specific. $d(u, v)$ is the word2vec embedding distance between

Table 2: Top-scoring text evidence for Qfact candidate (*Allianz Arena, 75000, "Capacity"*).

Evidence Qfact	Type	Source
(*Allianz Arena, 75000, "capacity, now"*)	Exact entity	In January 2015, a proposal to increase the capacity was approved by the city council so now Allianz Arena has a capacity of 75,000 (70,000 in Champions League).
(*Wembley Stadium, 90000, "official, capacity"*)	Type-based	It was revealed today that I have made an offer to purchase Wembley Stadium from The Football Association. ... The stadium opened in 2007 and has an official capacity of 90,000.
(*Great American Ball Park, 42271, "capacity"*)	Type-based	Great American Ball Park opened in 2003 at the cost of $290 million and has a capacity of 42,271.

Algorithm 1: Consistency-based Re-scoring

Input : Candidate Qfacts $\mathbb{F} = \{\mathcal{F}_1, \mathcal{F}_2, ... | \mathcal{F}_i = (e_i, q_i, X_i)\}$ with initial scores for Qquery $Q = (qt, qq, qX)$

Output: Consistency-aware scores of candidate Qfacts

1. Sample randomly a probe set from the candidate list $\mathbb{P} \subset \mathbb{F}$.
2. Train a Qfact quality predictor from the remaining candidate Qfacts $\mathbb{F} \setminus \mathbb{P}$, using initial scores as ground-truth.
3. Run the learned predictor on the probe set $\mathbb{P}$ to compute quality scores for all Qfacts in $\mathbb{P}$.
4. Repeat steps 1-3 a large number of times.
5. The *consistency score* of a candidate Qfact, *cons-score*($\mathcal{F}_i$), is computed as the average quality predicted, aggregated over all cases where $\mathcal{F}_i$ was in the probe set.

two words u (from the query) and v (from the answer candidate). In essence, this directed scoring function finds for each Qquery context word u the best matching token v from the Qfact context, taking context type into account by using $\sigma(v)$.

5.2 Corroboration by Inter-Fact Consistency

We use the *w-ded* distances between candidate answers and the query as *initial scores* for answer ranking. This initial ranking is further improved by considering the *mutual consistency* among the answer facts for the same query. To this end, we can exploit that our data often yields several Qfacts for the same answer entity. If all or most of them agree on their quantities and contextual cues, their scores should be close to each other. This idea can be generalized to all answer candidates even if they differ in their entities: they should still mostly agree on their contextual cues, and their quantities should have comparable order of magnitude. For example, if the candidate pool for answering a query about "British football stadiums with a seating capacity above 50,000" includes spurious results like (*Wembley Stadium, 32,000,000, "world cup, 1966, TV viewers"*), or (*Maracana Stadium, 78,838, "FIFA, Rio, 2014"*), these stand out against many good results by having the wrong order of magnitude in quantities or by missing important contextual cues about UK.

To detect and leverage such situations for elimination or demotion of noisy results, we have devised a method for consistency-aware corroboration and re-scoring of answer candidates. This is inspired by earlier work on consistency learning for image classification [39]. Algorithm 1 outlines this method.

The method is a form of self-validation, analogous to the principle of cross-validation. We randomly sample a probe set from the candidate Qfacts, and use the remaining Qfacts and their initial scores as ground-truth for training a quality predictor. The learned predictor is applied to the probe set, and we keep track of the predicted quality scores *cons-score*($\mathcal{F}_i$).

The difference between the initial score and the consistency score $|w\text{-}ded - cons\text{-}score|$ denotes the confidence of the initial score. A high difference between them denotes a noisy Qfact in the candidate list (i.e., either a high-ranked bad-Qfact, or low-ranked good-Qfact), which requires re-scoring.

Definition [Re-Scoring of Qfacts]. We re-score candidate fact $\mathcal{F}$ with regard to a query as a weighted combination of initial score (using *w-ded*) and consistency-aware *cons-score*:

$$final\text{-}score(\mathcal{F}) = (1 - \rho) \cdot w\text{-}ded(\mathcal{F}) \; + \; \rho \cdot cons\text{-}score(\mathcal{F})$$

with hyper-parameter ρ to control the re-scoring effect.

Learned Predictors. As this method requires frequent re-training of the predictor, we choose a very simple k-NN technique, which computes *cons-score* as the average initial scores of the k nearest Qfacts in the training set. This avoids the bottleneck of explicit re-training. We define the distance between Qfacts as the weighted combination of (1) the relative numeric distance between quantities (converted to standard units) and (2) the context similarity. The latter is computed by a vector space model, with features comprising the *tf-idf* values of context terms weighted by the context item from which they originate (column header, table caption, etc.).

6 EVALUATION

6.1 Intrinsic Evaluation of QuTE Components

We present experimental results on the key components of Qfact extraction: entity-quantity column alignment (CA) and entity linking (EL). The contextualization of Qfacts and the inter-fact consistency model matter only at query-time, and are thus evaluated in that extrinsic use case in Section 6.2.

Hyper-Parameter Tuning. Our method has a number of hyper-parameters for Qfact extraction: λ in Equation 1; weights for different context categories; and weights for the text-based evidence scoring model. For tuning these, we performed a grid search to determine the configuration with the best performance on a withheld validation dataset.

Testsets. Our experiments use three table collections:

- *Wiki_Links-Random:* a dataset introduced by [3], sampling 3000 tables from Wikipedia. As we are only interested in tables that express quantity properties, we filter this data and obtain a set of 259 tables, referred to as *Wiki_Links-Random_Qt*.
- *Equity:* a set of 69 content-rich tables introduced by [19]. Analogously to *Wiki_Links-Random*, we filter for tables with quantities, which results in a set of 30 tables, called *Equity_Qt*.
- *Wiki_Diff:* We observe that many tables from the above two datasets are easy cases for column alignment. Very often, the linked E-column is the first one, or the table has only one E-column, so linking all Q-columns to that one is trivially correct. Hence, we compile a new dataset called *Wiki_Diff*, consisting

Table 3: Column alignment precision *(macro_avg)*.

Method	*Wiki_L-R_Qt*	*Equity_Qt*	*Wiki_Diff*
Leftmost	0.736	0.817	0.045
Most-Unique	0.868	0.873	0.409
Closest-Left	0.728	0.674	0.705
Classifier [37]	0.864	0.717	0.597
Iterative CA	0.934	0.900	0.769

Table 4: Entity linking precision *(micro_avg)*.

Method	*Wiki_L-R_Qt*	*Equity_Qt*	*Wiki_Diff*
Prior	0.849	0.821	0.846
EL-MRF [3]	0.893	0.863	0.902
Joint EL&CA	0.900	0.876	0.902

of 134 Wikipedia tables, which are difficult cases for column alignment: there are at least two E-columns and the referred E-column is not the first one, or different Q-columns refer to different E-columns.

All three datasets originally contain only ground truth for entity linking; we annotated them with the proper column alignment.

Performance Metrics. For the CA task, we use the precision of correct alignments, macro-averaged over tables. Since there are many tables where all Q-columns refer to the same E-column, macro-averaging is meaningful to give each table the same weight (regardless of its width). For entity linking (EL) the metric is the precision, micro-averaged over entity mentions.

Results for Column Alignment. We compare our *Iterative CA* method with text-based evidence against several baselines (see Section 3.1): *(1) Leftmost*, *(2) Most-Unique*, *(3) Closest-Left*, and a *(4) Classifier* with features from column-wise properties (column-pair distances, distinct values per column, etc.) as employed by [37].

The results are shown in Table 3. We observe that our *Iterative CA* method outperforms all baselines by a large margin over all three datasets. This gives our approach a decisive advantage in extracting more and better quantity facts from web tables.

Results for Entity Linking. Although our EL method mostly follows prior works [3, 19], we report the performance of EL when computing jointly with CA, against two baselines: *(1) Prior* uses popularity of mention-entity pairs to link each mention to the most salient entity that matches the name, and *(2) EL-MRF* [3] is a state-of-the-art method based on MRF that incorporates priors, context similarity, row-wise coherence and column-wise coherence, but does not consider CA.

Table 4 shows that the *Joint EL&CA* method is as good as and sometimes better than the baselines, on all three datasets. Although the improvement over *EL-MRF* is not that large, it is notable and shows the positive impact of integrating CA information on the inference of EL.

Ablation Study. To analyze the influence of different components of our CA method, we conducted a comprehensive ablation study, by selectively disabling the following components: (1) type-based evidence for text-based scoring, and (2) coreference resolution for entity mentions when building the background Qfact collection from text. The results are shown in Table 5.

Table 5: Ablation study results for CA.

Method	*Wiki_L-R_Qt*	*Equity_Qt*	*Wiki_Diff*
Iterative CA	0.934	0.900	0.769
– type-based evidence	0.796	0.617	0.254
– coreferences	0.877	0.892	0.728

Table 6: Table collection statistics.

Source	*#tables*	*#E-Q-tables*	*#distinct-entities*	*#qfacts*
Wikipedia	1.8M	339K	757K	8.87M
TableL	2.6M	278K	255K	9.94M
Total	4.4M	618K	863K	18.81M

We observe that without type-lifted evidences, the CA precision decreases by more than 10 percent on all three datasets; for the difficult dataset *Wiki_Diff*, performance even drops by 50 percent. Exact-entity matching alone is insufficient as it suffers from the sparseness. This emphasizes the decisive role of our novel contribution to leverage external text evidence, as opposed to prior works that restricted information extraction from web tables to the tables themselves (and their local context). Disabling coreference resolution, for collecting background Qfacts from text, also degrades precision, but to much lesser degree: 5 percent at most.

6.2 Extrinsic Evaluation of Search and Ranking

This section presents experimental results for an end-to-end use case of quantity queries and their result rankings.

Hyper-Parameter Tuning. Analogously to Section 6.1, we tune query-time hyper-parameters for the *w-ded* distance and for the mixture with *cons-score* (see Section 5.2) by grid search for best Precision@10 on a withheld validation set.

Datasets. We run queries on a large collection of web tables compiled from two major sources:

- *TableL:* introduced by [20]. It contains 2.6M tables from 1.5M web pages, mostly falling under five major topics: finance, environment, health, politics, and sports.
- *Wikipedia Tables:* first introduced by [3]. As the original collection from 2015 is outdated, we processed a recent version of the English Wikipedia XML dump (March 2020) to construct an analogous dataset, containing a total of 1.8M tables.

The combined collection was filtered for tables that contain both E-columns and Q-columns. Table 6 shows data statistics of the large scale extraction, where we report the number of filtered E-Q-tables, the number of extracted Qfacts, and the number of extracted entities for each table corpus. In total, we end up with 618K tables and 18.8M extracted Qfacts, ready for large scale search.

Query Benchmarks. We use two sets of telegraphic queries:

- *Q100:* an established benchmark of 100 quantity queries from [16], featuring questions on a range of quantity measures for four domains: *Finance*, *Transport*, *Sports* and *Technology*. Ground-truth answers are annotated as *relevant* or *irrelevant* for the top-10 results of the original, text-based work in [16]. We extend these annotations to the top-10 results of all methods under comparison. However, there is no ground-truth about ideal top-10 results, like lists of all answers or answers sorted in ascending

or descending order of quantity value (e.g., the largest stadiums for a query about "sports arenas with capacity above 50K"). So there is no way to evaluate recall with this benchmark.

- *NewQ150:* To allow evaluating both precision and recall, we constructed a new collection of 150 queries, similar in nature to those of Q100 but such that each of them has a ground-truth answer list. To this end, we identified Wikipedia list pages that either capture the desired query result or provide a superset that is sorted by the quantity of interest. Examples for this kind of ground-truth is a list of all sprinters who ran 100 meters under 10 seconds, which by its sorting, also provides a sub-list of results under 9.9 seconds.

Performance Metrics. For both benchmarks, we report *Precision@10*, macro-averaged over 100 or 150 queries, respectively. For NewQ150, we also report *Recall@10* and *mAP@10* with regard to the answers in the ground-truth list.

Baselines. To the best of our knowledge, QuTE is the first system addressing quantity filters based on web tables. Therefore, there is no direct reference baseline; instead we compare against two strong baselines on quantity search over textual and general web contents:

- *Qsearch* is a text-based quantity search engine [16, 17] (accessible at *https://qsearch.mpi-inf.mpg.de/*). It runs on a collection of 21.7M Qfacts automatically extracted from sentences in Wikipedia articles and news articles from the New York Times archive and web crawls.

  This setup is not comparable to our QuTE method, as the underlying data sources are not the same. Nevertheless, having this baseline gives insights on the value of tapping into web tables.
- *Google* serves as the reference point for search-engine methodology. When we pose our benchmark queries, Google returns ten blue links along with preview snippets. The results are typically a mix of highly informative snippets, irrelevant snippets, and links to authoritative lists. These list pages often contain very good results, but the user would have to explicitly access and browse them (as opposed to being provided with direct answers in terms of entities).

For Google results, we assess the top-10 answer quality (with regard to the ground-truth top-10) in two different modes:

- *Direct answers (Google-DA):* only named entities that appear in the preview snippets are considered. This is a conservative mode, assuming lazy users who do not engage on further browsing.
- *List expansion (Google-LE):* each list-page answer (with the word "list" in its title) is fetched to materialize the list of entities, in the order of the list itself. Conceptually, this is done for each top-10 result of this kind, and the resulting lists are concatenated. The top-10 entities are considered as query answers in this mode, where users continue browsing.

Main Results. The precision results for the *Q100* benchmark are shown in Table 7. We see that Qsearch performs best for the top rank alone, but drops in precision with more results. This is because it is designed to retrieve a few high-confidence results and has very limited recall due its data based on single sentences. QuTE has lower precision but keeps this fairly high also for lower ranks, being able to find more correct answers from its table collection. The weak results for Google-DA show that search engines are really missing the ability to compute direct answers for quantity filters. Google-LE performs better, benefitting from list expansion because it often has one or two good super-lists of proper results in its 10 "blue links".

Table 7: Performance results for *Q100*.

System	*Prec.@1*	*Prec.@5*	*Prec.@10*
Google-DA	0.340	0.280	0.274
Google-LE	0.460	0.518	0.462
Qsearch	0.690	0.559	0.492
QuTE	0.540	0.512	0.491

Table 8: Performance results for *NewQ150*.

System	*Prec.@10*	*Recall@10*	*mAP@10*
Google-DA	0.167	0.076	0.041
Google-LE	0.342	0.251	0.193
Qsearch	0.290	0.177	0.119
QuTE	0.519	0.341	0.294

The results for the *NewQ150* benchmark are shown in Table 8, including recall and mAP for the top-10 query results. Here we see that QuTE clearly outperforms all baselines, especially in terms of recall@10 and mAP@10. Extracting Qfacts from web tables with high yield enables QuTE to compute many correct answers. Qsearch is limited by its text-based pool of candidate answers. The search engine again shows its missing support for quantity filters in direct-answers mode; in list-expansion mode, it performs much better but is still inferior to QuTE.

Table 9 shows a few anecdotal query results obtained by QuTE.

Ablation Study. To obtain insight into which components contribute how much, we performed an in-depth ablation study, by (1) discarding table-context categories from the *contextualization* step: dropping table captions, page titles, etc., except the Q-column header which was always kept as the most vital cue, and (2) disabling the inter-fact corroboration phase. The results of this study are shown in Tables 10 and 11 for Q100 and NewQ150 benchmarks, respectively.

We observe that page titles are the most important element for the contextualization step; discarding them led to a substantial drop in performance. As for the other context categories, their disabling resulted in some performance fluctuation, but overall their influence is relatively minor. So the bottom line is that page titles and column headers are crucial for Qfact extraction, and additional context categories do not have substantial benefits due their inherent noise.

The results also show that the inter-fact consistency corroboration is a vital component that improves the quality of top-10 results. Though the improvement is small (ca. 2 percent), the p-value from a paired t-test suggests that this improvement is statistically significant (0.034 and 0.019 for Q100 and NewQ150 benchmarks).

Text-based vs. Table-based Search. In terms of precision, Table 7 suggests that table-based (QuTE) and text-based query answering (Qsearch) produce results of comparable quality. Does that imply that they are interchangeable? However, a closer analysis shows that they are not simply interchangeable, but rather return complementary results. For Q100, each of the two methods has about 45% unique answers in their correct top-10 (not found by the other

Table 9: Anecdotal examples of quantity queries and top results by QuTE.

Query	Top Results
Skyscrapers higher than 1000 feet	Empire State Building, One World Trade Center, The Shard, Chrysler Building, etc.
British football teams worth more than 1.5 billion pounds	Manchester United F.C., Arsenal F.C., Liverpool F.C., Chelsea F.C., Manchester City F.C.
Sprinters who ran 100 meters under 9.9s	Usain Bolt, Carl Lewis, Maurice Greene, Justin Gatlin, Christian Coleman, etc.
Mobile games with number of players more than 250 million	Angry Birds, Super Mario Run, Candy Crush Saga, Temple Run, Pokémon Go, etc.

Table 10: Ablation study results on *Q100*.

Method	*Prec.@1*	*Prec.@5*	*Prec.@10*
QuTE	0.540	0.512	0.491
– page title	0.450	0.438	0.421
– table caption	0.550	0.522	0.477
– same-row cells	0.520	0.502	0.485
– dom-tree headings	0.540	0.504	0.486
– surrounding text	0.560	0.504	0.481
– corroboration	0.530	0.494	0.475

Table 11: Ablation study results on *NewQ150*.

Method	*Prec.@10*	*Recall@10*	*mAP@10*
QuTE	0.519	0.341	0.294
– page title	0.434	0.286	0.233
– table caption	0.495	0.327	0.277
– same-row cells	0.513	0.338	0.295
– dom-tree headings	0.521	0.341	0.293
– surrounding text	0.513	0.336	0.289
– corroboration	0.497	0.327	0.279

method). For NewQ150, which tends to have more difficult queries, the fractions are 18% for QuTE and 8% for Qsearch.

7 RELATED WORK

Quantity Recognition. Detecting quantities in text and tables has been well researched, with prevalent methods based on rules, CRFs or neural learning [1, 19, 25, 31–33]. This involves recognizing numeric expressions in combination with units, and ideally includes also normalization of values (considering scale indicators as in "10 mio" or "10K") and conversions of units (e.g., from US dollars into GBP or MPG-e into kWh/100km). Normalization and conversions are handled via rules. This prior work solely focuses on the numeric quantity alone, and does not include inferring to which entity the quantity refers. Moreover, it does not identify contextual cues that are necessary for querying. Our paper starts with state-of-the-art quantity recognition, and makes novel contributions on inferring respective entities and relevant contexts.

Fact Extraction from Web Tables. Ad-hoc tables in HTML pages and spreadsheet contents have been studied as a target for entity and concept linking, fact extraction, search and question answering. The surveys by [6] and [41] discuss the relevant literature.

Our work builds on state-of-the-art entity linking for web tables [3, 14, 19, 23, 29] sharing the general approach of combining per-row contexts with per-column coherence based on probabilistic graphical models or random walks.

Prior methods for fact extraction from tables, for the task of KB augmentation, have followed the standard model of SPO triples, with focus on entity linking for the S and O arguments from the same row [10, 15, 21, 26, 29, 30]. Target predicates P are assumed to come from a pre-existing knowledge base (as opposed to OpenIE). None of the prior works distinguish whether the O column contains entities or numeric quantities. In contrast, our method includes specific techniques to handle quantity columns.

A prevalent assumption is that there is a single subject column where all S arguments come from, regardless of the choice of O column. Some works use the heuristics that S is the leftmost non-numeric column of a table; other works employ a supervised classifier based on simple features of candidate columns [8, 37]. Our approach does not make this assumption of a single subject column, thus being able to tap into more complex content-rich tables. The only prior work that considered multiple S-columns is [5]. This method critically relies on the detection of approximate functional dependencies and value correlations between column pairs. This does not work for quantity columns, though, as their values can be anywhere between all-distinct and many-duplicates (e.g., if stadium capacities in Table 1 were crudely rounded to 50K, 60K, etc.).

Entity Search and Question Answering. Entity-centric search and question answering are broad areas that cover a variety of information-seeking needs, see surveys like [2, 9, 18, 28]. As far as quantities are concerned, lookups are supported by many methods, over both knowledge graphs and text documents, and are part of major benchmarks, such as QALD [36], NaturalQuestions [22], ComplexWebQuestions [35], LC-QuAD [12] and others. However, lookups such as "What is the value of Real Madrid?" or "energy consumption of Toyota Prius Prime" are much easier to process than queries with quantity filters. The former do not need to interpret quantities in terms of measure, value and unit, whereas this is crucial for evaluating filter conditions. The only prior work that specifically addressed quantity filters is [16, 17], which was solely based on textual contents, though.

Search and QA over web tables have been addressed in various settings. Methods in [7, 27, 33, 40] support querying heterogeneous collections of tables, but focus on the joint mapping of keywords onto entities and column headers in the underlying data. Quantity-filter queries are not addressed.

8 CONCLUSION

This paper presents the first method, called QuTE, for extracting quantity facts from web tables, to support queries with quantity filters. In experiments, QuTE clearly outperforms both prior works on text-based Qfacts and a major search engine. An overarching goal of this work is to extensively populate a high-quality knowledge base with quantity properties, including advanced measures such as energy consumption and carbon footprint for car models. This is ongoing and future work.

REFERENCES

[1] Omar Alonso and Thibault Sellam. 2018. Quantitative Information Extraction From Social Data. In *The 41st International ACM SIGIR Conference on Research & Development in Information Retrieval, SIGIR 2018, Ann Arbor, MI, USA, July 08-12, 2018.*

[2] Krisztian Balog. 2018. *Entity-Oriented Search.* The Information Retrieval Series, Vol. 39. Springer.

[3] Chandra Sekhar Bhagavatula, Thanapon Noraset, and Doug Downey. 2015. TabEL: Entity Linking in Web Tables. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I.*

[4] Alexander Bondarenko, Pavel Braslavski, Michael Völske, Rami Aly, Maik Fröbe, Alexander Panchenko, Chris Biemann, Benno Stein, and Matthias Hagen. 2020. Comparative Web Search Questions. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020.*

[5] Katrin Braunschweig, Maik Thiele, and Wolfgang Lehner. 2015. From Web Tables to Concepts: A Semantic Normalization Approach. In *Conceptual Modeling - 34th International Conference, ER 2015, Stockholm, Sweden, October 19-22, 2015, Proceedings.*

[6] Michael J. Cafarella, Alon Y. Halevy, Hongrae Lee, Jayant Madhavan, Cong Yu, Daisy Zhe Wang, and Eugene Wu. 2018. Ten Years of WebTables. *Proc. VLDB Endow.* 11, 12 (2018).

[7] Kaushik Chakrabarti, Zhimin Chen, Siamak Shakeri, and Guihong Cao. 2020. Open Domain Question Answering Using Web Tables. *CoRR* abs/2001.03272 (2020). arXiv:2001.03272

[8] Dong Deng, Yu Jiang, Guoliang Li, Jian Li, and Cong Yu. 2013. Scalable Column Concept Determination for Web Tables Using Large Knowledge Bases. *Proc. VLDB Endow.* 6, 13 (2013).

[9] Dennis Diefenbach, Vanessa López, Kamal Deep Singh, and Pierre Maret. 2018. Core techniques of question answering systems over knowledge bases: a survey. *Knowl. Inf. Syst.* 55, 3 (2018).

[10] Xin Dong, Evgeniy Gabrilovich, Geremy Heitz, Wilko Horn, Ni Lao, Kevin Murphy, Thomas Strohmann, Shaohua Sun, and Wei Zhang. 2014. Knowledge vault: a web-scale approach to probabilistic knowledge fusion. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014.* ACM.

[11] Xin Luna Dong, Hannaneh Hajishirzi, Colin Lockard, and Prashant Shiralkar. 2020. Multi-modal Information Extraction from Text, Semi-structured, and Tabular Data on the Web. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: Tutorial Abstracts, ACL 2020, Online, July 5, 2020.*

[12] Mohnish Dubey, Debayan Banerjee, Abdelrahman Abdelkawi, and Jens Lehmann. 2019. LC-QuAD 2.0: A Large Dataset for Complex Question Answering over Wikidata and DBpedia. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II.*

[13] Julian Eberius, Katrin Braunschweig, Markus Hentsch, Maik Thiele, Ahmad Ahmadov, and Wolfgang Lehner. 2015. Building the Dresden Web Table Corpus: A Classification Approach. In *2nd IEEE/ACM International Symposium on Big Data Computing, BDC 2015, Limassol, Cyprus, December 7-10, 2015.* IEEE Computer Society.

[14] Vasilis Efthymiou, Oktie Hassanzadeh, Mariano Rodriguez-Muro, and Vassilis Christophides. 2017. Matching Web Tables with Knowledge Base Entities: From Entity Lookups to Entity Embeddings. In *The Semantic Web - ISWC 2017 - 16th International Semantic Web Conference, Vienna, Austria, October 21-25, 2017, Proceedings, Part I.*

[15] Besnik Fetahu, Avishek Anand, and Maria Koutraki. 2019. TableNet: An Approach for Determining Fine-grained Relations for Wikipedia Tables. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019.*

[16] Vinh Thinh Ho, Yusra Ibrahim, Koninika Pal, Klaus Berberich, and Gerhard Weikum. 2019. Qsearch: Answering Quantity Queries from Text. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part I (Lecture Notes in Computer Science, Vol. 11778).* Springer.

[17] Vinh Thinh Ho, Koninika Pal, Niko Kleer, Klaus Berberich, and Gerhard Weikum. 2020. Entities with Quantities: Extraction, Search, and Ranking. In *WSDM '20: The Thirteenth ACM International Conference on Web Search and Data Mining, Houston, TX, USA, February 3-7, 2020.*

[18] Zhen Huang, Shiyi Xu, Minghao Hu, Xinyi Wang, Jinyan Qiu, Yongquan Fu, Yuncai Zhao, Yuxing Peng, and Changjian Wang. 2020. Recent Trends in Deep Learning Based Open-Domain Textual Question Answering Systems. *IEEE Access* 8 (2020).

[19] Yusra Ibrahim, Mirek Riedewald, and Gerhard Weikum. 2016. Making Sense of Entities and Quantities in Web Tables. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016.*

[20] Yusra Ibrahim, Mirek Riedewald, Gerhard Weikum, and Demetrios Zeinalipour-Yazti. 2019. Bridging Quantities in Tables and Text. In *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019.* IEEE.

[21] Benno Kruit, Peter A. Boncz, and Jacopo Urbani. 2019. Extracting Novel Facts from Tables for Knowledge Graph Completion. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part I.*

[22] Tom Kwiatkowski, Jennimaria Palomaki, Olivia Redfield, Michael Collins, Ankur P. Parikh, Chris Alberti, Danielle Epstein, Illia Polosukhin, Jacob Devlin, Kenton Lee, Kristina Toutanova, Llion Jones, Matthew Kelcey, Ming-Wei Chang, Andrew M. Dai, Jakob Uszkoreit, Quoc Le, and Slav Petrov. 2019. Natural Questions: a Benchmark for Question Answering Research. *Trans. Assoc. Comput. Linguistics* 7 (2019).

[23] Girija Limaye, Sunita Sarawagi, and Soumen Chakrabarti. 2010. Annotating and Searching Web Tables Using Entities, Types and Relationships. *Proc. VLDB Endow.* 3, 1 (2010).

[24] Qing Lu and Lise Getoor. 2003. Link-based Classification. In *Machine Learning, Proceedings of the Twentieth International Conference (ICML 2003), August 21-24, 2003, Washington, DC, USA.*

[25] Aman Madaan, Ashish Mittal, Mausam, Ganesh Ramakrishnan, and Sunita Sarawagi. 2016. Numerical Relation Extraction with Minimal Supervision. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA.*

[26] Yaser Oulabi and Christian Bizer. 2019. Extending Cross-Domain Knowledge Bases with Long Tail Entities using Web Table Data. In *Advances in Database Technology - 22nd International Conference on Extending Database Technology, EDBT 2019, Lisbon, Portugal, March 26-29, 2019.*

[27] Rakesh Pimplikar and Sunita Sarawagi. 2012. Answering Table Queries on the Web using Column Keywords. *Proc. VLDB Endow.* 5, 10 (2012).

[28] Ridho Reinanda, Edgar Meij, and Maarten de Rijke. 2020. Knowledge Graphs: An Information Retrieval Perspective. *Found. Trends Inf. Retr.* 14, 4 (2020).

[29] Dominique Ritze and Christian Bizer. 2017. Matching Web Tables To DBpedia - A Feature Utility Study. In *Proceedings of the 20th International Conference on Extending Database Technology, EDBT 2017, Venice, Italy, March 21-24, 2017.*

[30] Dominique Ritze, Oliver Lehmberg, and Christian Bizer. 2015. Matching HTML Tables to DBpedia. In *Proceedings of the 5th International Conference on Web Intelligence, Mining and Semantics, WIMS 2015, Larnaca, Cyprus, July 13-15, 2015.* ACM.

[31] Subhro Roy, Tim Vieira, and Dan Roth. 2015. Reasoning about Quantities in Natural Language. *Transactions of the Association for Computational Linguistics* 3 (2015).

[32] Swarnadeep Saha, Harinder Pal, and Mausam. 2017. Bootstrapping for Numerical Open IE. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 2: Short Papers.*

[33] Sunita Sarawagi and Soumen Chakrabarti. 2014. Open-domain quantity queries on web tables: annotation, response, and consensus models. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA - August 24 - 27, 2014.*

[34] Wei Shen, Jianyong Wang, and Jiawei Han. 2015. Entity Linking with a Knowledge Base: Issues, Techniques, and Solutions. *IEEE Trans. Knowl. Data Eng.* 27, 2 (2015).

[35] Alon Talmor and Jonathan Berant. 2018. The Web as a Knowledge-Base for Answering Complex Questions. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers).*

[36] Christina Unger, Corina Forascu, Vanessa López, Axel-Cyrille Ngonga Ngomo, Elena Cabrio, Philipp Cimiano, and Sebastian Walter. 2015. Question Answering over Linked Data (QALD-5). In *Working Notes of CLEF 2015 - Conference and Labs of the Evaluation forum, Toulouse, France, September 8-11, 2015.*

[37] Petros Venetis, Alon Y. Halevy, Jayant Madhavan, Marius Pasca, Warren Shen, Fei Wu, Gengxin Miao, and Chung Wu. 2011. Recovering Semantics of Tables on the Web. *Proc. VLDB Endow.* 4, 9 (2011).

[38] Zhibiao Wu and Martha Stone Palmer. 1994. Verb Semantics and Lexical Selection. In *32nd Annual Meeting of the Association for Computational Linguistics, 27-30 June 1994, New Mexico State University, Las Cruces, New Mexico, USA, Proceedings.*

[39] Jay Yagnik and Atiq Islam. 2007. Learning people annotation from the web via consistency learning. In *Proceedings of the 9th ACM SIGMM International Workshop on Multimedia Information Retrieval, MIR 2007, Augsburg, Bavaria, Germany, September 24-29, 2007.*

[40] Mohamed Yakout, Kris Ganjam, Kaushik Chakrabarti, and Surajit Chaudhuri. 2012. InfoGather: entity augmentation and attribute discovery by holistic matching with web tables. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2012, Scottsdale, AZ, USA, May 20-24, 2012.*

[41] Shuo Zhang and Krisztian Balog. 2020. Web Table Extraction, Retrieval, and Augmentation: A Survey. *ACM Trans. Intell. Syst. Technol.* 11, 2 (2020).