Bridging Quantities in Tables and Text

Yusra Ibrahim
Max Planck Institute for Informatics
Saarbrücken, Germany
yibrahim
@mpi-inf.mpg.de

Mirek Riedewald
Northeastern University
Boston, USA
m.riedewald
@northeastern.edu

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

Demetrios Zeinalipour-Yazti
University of Cyprus
Nicosia, Cyprus
dzeina@cs.ucy.ac.cy

***Abstract*—There is a wealth of schema-free tables on the Web, holding valuable information about quantities on sales and costs, environmental footprint of cars, health data and more. Table content can only be properly interpreted in conjunction with the textual context that surrounds the tables. This paper introduces the *quantity alignment problem*: bidirectional linking between textual mentions of quantities and the corresponding table cells, in order to support advanced content summarization and faster navigation between explanations in text and details in tables. We present the *BriQ* system for computing such alignments. BriQ is designed to cope with the specific challenges of approximate quantities, aggregated quantities, and calculated quantities in text that are common but cannot be directly matched in table cells. We judiciously combine feature-based classification with joint inference by random walks over candidate alignment graphs. Experiments with a large collection of tables from the Common Crawl project demonstrate the viability of our methods.**

I. Introduction

Tables not just epitomize relational databases, but are also widely used to represent data on the Web (embedded in HTML pages) and in enterprises (in spreadsheets). Unlike in databases, these tables are often created in an ad-hoc manner, without proper schema design and with highly heterogeneous formats of attribute values. Therefore, the interpretation of tables, by human analysts and other users, often hinges on additional text that discusses the table content.

Figure 1 shows excerpts of Web pages from the domains of health, environment and finance. The currency of the financial numbers in Figure 1c becomes clear only when reading the text. Likewise, it is the text of Figure 1b that points the user to the most expensive of the three cars.

To make sense of tables, it is thus crucial that table rows, columns and individual cells are connected with relevant snippets in the surrounding text. For entire rows and for cells with names of products, companies, locations, etc., this is the problem of *entity linking* [33]. Specific methods for tables as input have been developed [2], [18], [19]. However, this does not capture the *quantities* in individual cells. Linking quantities has been addressed in [13], [31], but these works assume that a knowledge base or reference system of canonicalized quantities (with standardized measures, proper units, etc.) is available. In practice, knowledge bases for quantities are merely small and limited to special domains.

In this paper, we aim to link quantities without making such assumptions. We do so by linking table cells with relevant pieces of the text that accompanies a table. This supports users in two ways. First, in going from tables to text, they obtain explanations of the mere numbers in cells and their relevance for the topic at hand. Second, in going from text to tables, the user can drill down on statements in terms of detailed numbers. Figure 1 illustrates these benefits by the overlaid bidirectional edges. Quantity alignment links the text to data from the tables, and vice versa. Hence, it can be combined with entity linking techniques to augment knowledge bases. Furthermore, quantity alignment creates an opportunity for advanced automatic text summarization [10], [22], which currently does not include table data. Once our system identifies aligned quantities, it is possible to determine which table rows, columns, and individual cells are referenced by the text summary—so that they can be added to it. And since our approach distinguishes between simple single-cell references and aggregates, it can provide hints to an automatic text summarizer. For instance, knowing that one sentence references a row sum, while another discusses individual values in the same row, the summarization algorithm could decide to include the former in the summary, but not the latter.

Problem Statement: We formalize the problem of bridging quantities in tables and text as a *quantity alignment problem*: For a text document with one or more tables,

- detect quantity mentions in text that refer to table cells
- and map these mentions to their proper cells.

Here, quantity mentions are textual expressions that contain numbers, but also include phrases that refer to aggregation, ranking and change rates. For example, in Figure 1a, the phrase "total of 123 patients" refers to an aggregate value, namely, the sum of the values in the sales column. In Figure 1b, "the least affordable option" refers to the maximum price in a column, and in Figure 1c, "increased by 1.5%" refers to the rate of change.

Although the problem resembles that of entity linking, it is more challenging (and unexplored) for several reasons:

- There is no explicit knowledge base that contains all targets (namely, entities) of the desired mapping. In our setting, the targets of the alignment are the values in table cells (often in incomplete or noisy formats), and the number of possible mention-cell pairs that could be aligned is huge.
- Quantity mentions in text often differ in their formats from their counterparts in table cells. For example, "37K EUR"

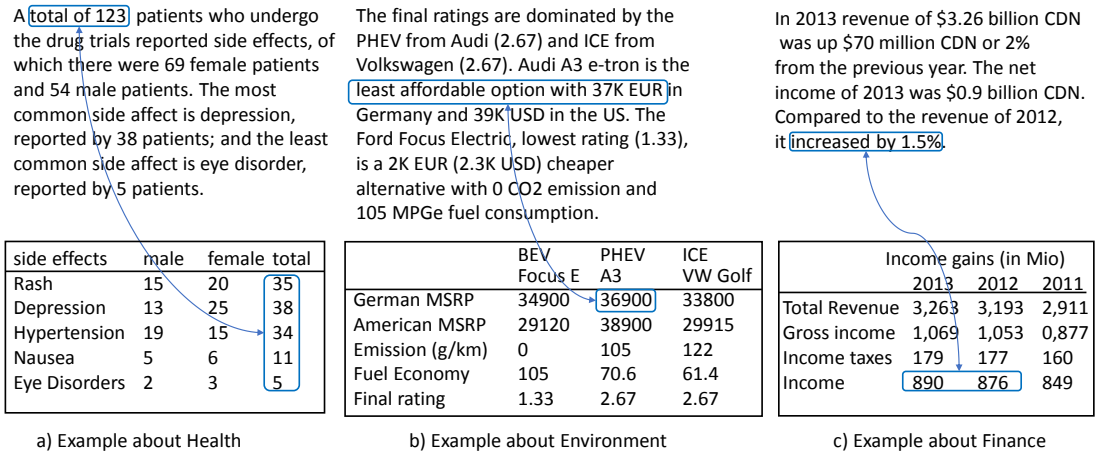

Fig. 1. Examples of Web Tables with Explanatory Texts

(in Figure 1b) refers to "36900" in a cell with row header "German MSRP" (in a rotated table). Such *approximate mentions* are frequent.

- *Aggregate quantities* that appear in text in forms such as "total of 123 patients" (Figure 1a) are not necessarily present in any table cell, but simply correspond to a column total. In such cases, the text mention should be aligned with all cells of the respective column to be summed up.
- Other forms of *calculated quantities* like maximum values, differences, change rates, etc., require alignments of text phrases like "least affordable" (Figure 1b), "up $70 million Cdn" (Figure 1c), "increased by 1.5%" (Figure 1c) etc. with a set of cells, typically in the same row or column.

Our Approach: For aligning quantity mentions in text with cells in tables, we have developed a full-fledged system called *BriQ* (for "*Bri*dging *Q*uantities in tables and text"). The core of *BriQ* is a hybrid algorithm for mapping mentions onto cells, by first learning a supervised classifier that accepts or drops mention-cell candidate pairs. The classifier not only serves to prune the search space, but also yields a prior for additional unsupervised steps based on random walks over appropriately weighted candidate graphs. The latter steps harness joint inference over the full alignment of all mentions in a document and all candidate cells in one or more tables within the document. To minimize dependence on hard-to-obtain training data and to cope with larger scale, the joint inference is unsupervised.

Our methods pay particular attention to the challenges of aggregated (e.g., column totals) and calculated quantities (e.g., change rates). We do this by carefully generating candidates in the form of "virtual cells," standing for cell combinations such as table columns or same-row cell pairs. For example, a virtual cell is generated for a column total even if the table itself does not explicitly show the total. We devise various techniques to prune the number of such virtual candidate cells, to ensure computational tractability and to control spurious matches.

Contributions: Salient points of this paper are:

- We introduce and formalize the novel problem of *quantity alignment* for Web pages that contain text and one or more ad-hoc tables.
- We present the *BriQ* system[1], including a two-stage algorithm for computing alignments, with a trained classifier as a prior and unsupervised, random-walk-based, techniques for global inference.
- Comprehensive experiments, with a large collection of Web tables and high-quality ground-truth annotations, demonstrate the practical viability of the *BriQ* method and its superior performance over two baselines.

II. System Overview

A. Computational Model

The *BriQ* method takes the following inputs:

- A piece of text, like a (part of a) web page, with a set of m text mentions of quantities $X = \{x_i : i = 1, \ldots, m\}$.
- A table q with r rows and c columns and a set of n mentions of quantities $T = \{t_j : j = 1, \ldots, n\}$.

Text mentions include terms containing numbers or numerals such as "123 patients", "37K EUR", "1.5%" or "twenty pounds". To focus on informative quantities, we eliminate date/time, headings (such as "Section 1.1"), phone numbers and references (such as "[2]", "Win10").

Table mentions include two types of quantities. The first are explicit **single-cell mentions**, such as '36900' in Table 1c, second row, third column. Given a table with r rows and c columns we have at most $r \cdot c$ single-cell quantity mentions. The second type of are **composite quantity mentions** (or *virtual-cell* mentions), computed as an aggregation of one or more table cells, such as '123', the sum for the fourth column in Table 1a.

[1]code and dataset available at:
https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/briq/

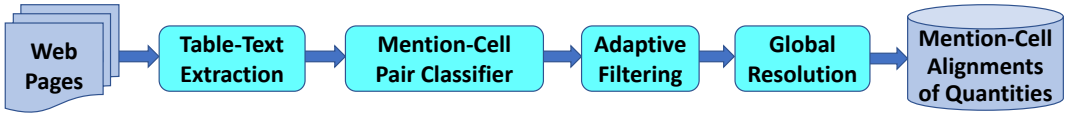

Fig. 2. *BriQ* System Architecture

We consider a broad range of **aggregate functions** that take two or more table cells as input and produce a single quantity:

- *Sum*: given q quantities, $\text{sum}(y_1, .., y_q) = \sum_{i=1}^{q} y_i$
- *Difference*: given 2 quantities, $\text{diff}(a, b) = a - b$
- *Percentage*: given 2 quantities, $\text{pct}(a, b) = \frac{a}{b} \cdot 100\%$
- *Change Ratio*: given 2 quantities, $\text{ratio}(a, b) = \frac{a-b}{a}$
- *Average*: given q quantities, $\text{avg}(y_1, ..., y_q) = \frac{\sum_{i=1}^{q} y_i}{q}$
- *Max* or *Min*: given q quantities, $\max_{i=1}^{q} y_i$ or $\min_{i=1}^{q} y_i$

These composite quantities may be present in a table already, but we also consider them if they are not explicit as the surrounding text may still refer to totals, diffs, etc. Hence the notion of *virtual-cell mentions*. In our experience, aggregates almost always refer to cells in the same row or column. More precisely, sum, average, min, and max tend to be computed for an entire row or column, resulting in $O(r + c)$ composite quantity candidates in the table. Since difference, percentage, and change ratio aggregate two values in a row or column, there are $O(\binom{r}{2} + \binom{c}{2})$ candidates for them. This leads to a quadratic (in table size) search space for the alignments, which is prohibitive for large tables. We will present adaptive filtering techniques for carefully pruning this search space.

Note that this model can be generalized by considering aggregations over other subsets of table cells, and even cells in different tables. For example, the text in Figure 1c could possibly refer to "the total income of the last two years," which is the sum of two cells (in the 2013 and 2012 columns) rather than a row total. With this generalization, the search space of the alignment problem would further increase, becoming exponential in table size already when arbitrary subsets of cells in a row or column are considered. The BriQ framework can handle this extended setting as well, and we studied it experimentally. It turned out, however, that such sophisticated cases are very rare, and hence did not have any impact on the overall quality of the BriQ outputs. For run-time efficiency, we consider only the case where sums and averages are restricted to entire rows or entire columns or two cells in the same row or same column, leaving the rare cases for future work.

The BriQ framework can handle a broad range of aggregation functions. However, in our experiments we only consider aggregations that appeared in at least 5% of the tables, because we need a sufficient number of examples to train and evaluate our models. Therefore, we only consider the following four aggregations in our experiments: sum, difference, percentage and change ratio.

For aligning quantity mentions between text and table, we aim to compute as output a subset of mention pairs $\langle x_i, t_j \rangle$ where $x_i \in X$ is a text mention and $t_j \in T$ is a table mention, including virtual cells for composite quantities. These pairs should denote the same quantity with high confidence. For the examples in Figure 1, the algorithm output should include the following pairs:

- $\langle$ "total of 123", sum('35','38','34','11','5') $\rangle$,
- $\langle$ "least affordable option with 37K EUR", '36900' $\rangle$,
- $\langle$ "increased by 1.5%", ratio('890','876') $\rangle$.

BriQ also returns the locations of the mentions, which we omitted here for the sake of presentation. Note that alignments include approximate values such as "37K EUR" and composite quantities that are not explicitly present in the table, such as 'ratio('890','876').' The alignment would ideally be a total mapping, covering all text mentions in the input. However, realistic cases may contain numbers in text that do not refer to any table—so we compute a partial mapping.

B. BriQ Architecture

Figure 2 gives a pictorial overview of the *BriQ* system architecture. In the following, we outline each of the shown components.

1) Table-Text Extraction: This module takes as input a web page and splits it into coherent segments, which we refer to as *documents*. Each document consists of a sequence of paragraphs and one or more tables to which the text refers. For each document, quantity mentions are extracted from the text and the tables, using regular expressions. Virtual cells—for aggregated quantities—are automatically generated by considering: (i) all rows and columns for totals; and (ii) all pairs of cells in the same row or column for difference, percentage, and change ratio.

2) Mention-Cell Pair Classification: This module first computes features for each text mention and each table mention by analyzing surrounding context. Also, similarity-based features are computed for each pair of text mention and table cell that could be a candidate pair for alignment. We use manually annotated web pages with ground-truth alignment to train a binary classifier that accepts or rejects candidate pairs.

The classifier operates *locally* in the sense that it predicts the alignment confidence for each mention-cell pair in isolation, i.e., it does not consider several mention-cell pairs together for joint inference. It serves two purposes: First, it enables the subsequent filtering step, which significantly reduces the number of candidate pairs. This is essential for achieving acceptable running time in the more expensive global resolution step. Second, it provides a prior for that global resolution step.

3) Adaptive Filtering: This stage filters the classifier's output to arrive at a sufficiently small set of candidate pairs that the subsequent global resolution can handle. The filtering uses the confidence scores of the classifier, but also considers more sophisticated measures to adapt to the specifics of different situations.

4) Global Resolution: This module takes as input the candidate mention-pairs from the classifier and outputs the final alignment of quantities between text and tables. It uses the classifier confidence values as prior weights, and employs global inference methods such as random walks over graphs to resolve the alignments.

III. Table-Text Extraction

Web pages, such as Wikipedia articles or product test reports, can be very long and cover a variety of thematic aspects, along with several tables. We therefore pre-process and split them into *coherent documents*. Since paragraphs form a natural unit in text for discussing a specific aspect, we use them as atomic building blocks. More precisely, we define a coherent document to be a paragraph together with all "related" tables from the same Web page. Related tables are identified by computing pairwise similarities between all paragraphs and all tables in the page, and then selecting those with similarity above a threshold. We consider tokens in the entire content of the table including column headers and captions. Note that a paragraph may have more than one related table, and a table might be related to multiple paragraphs.

For each document, we extract all quantity mentions from both text and tables, using regular expression matching (e.g., '`\d+\s*\p{Currency_Symbol}`' for monetary values). Quantities are extracted from text as follows: first we identify and remove complex quantities that involve multiple parts, such as '5 ± 1 km per hour'. Then, we extract simple quantities, such as '$500 million' and '1.34%'. This order ensures that complex quantities are not erroneously split into several matches. For tables, we employ the same procedure and attempt to extract a single quantity mention per cell, together with its unit (if present). In addition, we also attempt to extract information about the unit from each row and column header, footer, and the caption. We normalize quantity mentions; for example '0.5 million' transformed to '500 000'.

IV. Mention-Pair Classification

This stage of the *BriQ* system applies supervised learning to predict if a text mention does refer to a table mention so that they should be aligned. This binary classifier performs local resolution in the sense that it makes a prediction for an *individual* mention-pair, not taking into account dependencies between predictions made for *different* mention-pairs. Such couplings will be considered by the global resolution later, at much higher computational costs, however. The confidence scores of the classifier serve as prior weights for the joint inference at the global resolution stage.

A. Classification Algorithms

We use a **Random Forest (RF)** classifier for this purpose. RFs are among the most powerful classifiers that are not prone to overfitting. An RF classifier consists of an ensemble of decision trees, each trained on an independent bootstrap sample of the training data. The final prediction for an input is obtained based on the majority vote of the individual trees, returning the fraction of votes for the "related" class as the probability of the mention-pair being related. It has been shown that RFs yield well calibrated probabilities [5], [25], which is important for our usage of RF outputs fed into the global resolution stage.

B. Features

We judiciously designed a variety of features that capture information a human reader would use in order to determine if text mention x and table cell t denote the same quantity. The alternative—automatic representation learning, e.g., with Deep Learning—was not viable for our problem due to the limited amount of labeled data and the high cost for obtaining it (see Section VII). Overall, we believe that the complexity of our problem setting is better served by modeling informative features rather than solely relying on end-to-end learning with limited training data.

The most obvious basic feature is **surface form similarity**, $\mathrm{f}_1(x,t)$. We adopted the Jaro-Winkler distance measure to compute the string similarity between the surface form of the text mention against the table mention. We use Jaro-Winkler because it emphasizes a match at the beginning of the string, which is desirable when comparing quantity mentions. For example, a quantity mention "26.7$" in the text is closer to "26.65$" than to "29.75$".

B.1. Context Features

Local context word overlap, $\mathrm{f}_2(x,t)$, measures the similarity between the local contexts of a pair of text and table mention. A window of n words preceding and following the text mention is considered; for the table mention it is the full row and the full column content. The feature value is defined as the weighted overlap coefficient between the two bags of words. That is, we assign a weight to each word relative to its position. We use the following formula to compute the weight of a word e at distance d from the text mention:

$$\mathrm{weight}(e) = 1 - \left(\frac{d}{\mathrm{stepSize}} \cdot \mathrm{stepWeight} \right),$$

where $\mathrm{stepWeight}$ is the discounted weight at each $\mathrm{stepSize}$ away from the text mention. Then, we compute the overlap coefficient using these weights. We tune n, $\mathrm{stepSize}$, and $\mathrm{stepWeight}$ on the withheld validation dataset.

Global context word overlap, $\mathrm{f}_3(x,t)$, is similar to $\mathrm{f}_2(x,t)$, but uses the entire paragraph as the context of the text mention; and the entire table content as the context of the table mention.

Local context phrase overlap, $\mathrm{f}_4(x,t)$, measures the similarity between the *noun phrases* in the local context of text and table mention. The local context of the text mention is the sentence in which the text mention occurs; and for the table mention it is the full row and the full column content.

For example, the noun phrase "segment profit' in Figure 3.

Global context phrase overlap, $\mathrm{f}_5(x,t)$, is defined analogously, but considers noun phrases in the entire paragraph as the global context of the text mention; and the noun phrases in the entire table as the global context of the table mention.

B.2. Quantity Features

Relative difference between normalized quantity values, $\mathrm{f}_6(x,t) = \frac{|x-t|}{max(x,t)}$, reflects the numeric distance between mentions. Here, x and t denote the numerical values of the respective mentions, after normalization. In Figure 1 the normalized value of mention '37K EUR' is 37000.

Unnormalized relative difference between quantities, $\mathrm{f}_7(x,t)$, is the relative difference of the values *without* normalization. For example, the unnormalized value of mention '37K EUR' is 37.

Unit match, $\mathrm{f}_8(x,t)$, is a four-valued categorical feature that captures the degree to which the quantity units match. A *strong match* occurs when both mentions have a specified unit and these units match; a *weak match* when both mentions have no specified units; a *weak mismatch* when only one mention has a specified unit; and a *strong mismatch* when both mentions have a specified unit and these units do not match.

Scale difference, $\mathrm{f}_9(x,t)$, is the difference in the orders of magnitude between two quantities. For example, the scale difference of '37000' and '37' is 3 (powers of ten).

Precision difference, $\mathrm{f}_{10}(x,t)$, captures the difference in the number of digits after the decimal point. For example the precision difference of '1.5' and '1.543' is 2.

Approximation indicator, $\mathrm{f}_{11}(x,t)$, reflects if the text mention is accompanied by a modifier indicating an approximation. This categorical feature can take on values 'approximate', 'exact', 'upper bound', and 'lower bound'. These are derived from text cues like "ca.", "about", "nearly", "more than", etc.

Aggregate function match, $\mathrm{f}_{12}(x,t)$, is the degree to which the aggregate function for computing the value of the cell or virtual cell matches the kind of aggregation for the text mention as inferred from text cues. We implement this by looking up the words around the text mention in a dictionary that maps words to names of aggregate functions. (We set the neighborhood size by default to five words; but this could also be tuned on the validation data.) Analogous to the unit-match feature, there are four possible values: *strong match*, *weak match*, *weak mismatch*, and *strong mismatch*. For example in Figure 1(a) the inferred aggregation of mention 'total of 123 patients' is *sum* and it has a strong match with the aggregation of the virtual cell carrying the sum of the last column; and it has a strong mismatch with the virtual cell carrying the average of the last column.

V. Adaptive Filtering

As discussed in Section II, it is essential for performance to significantly reduce the number of mention-pair candidates for global resolution, typically from 1000s of candidates to 100s for tractability of global inference algorithms. An obvious approach for the necessary filtering would be to use the classifier's confidence scores: we could retain only candidates above a certain threshold, or we could keep a certain number of highest-scoring candidates. While superficially appealing, it is rather rigid and disregards the need to handle different kinds of quantity mentions in a more flexible way, e.g., simple quantities vs. aggregate quantities. Hence we devised an *adaptive filtering* strategy as follows. First we develop a *text mention tagger* to predict the aggregation function for each text mention, or tag the mention as a single-cell match. Then, we prune mention-pairs based on this tagger's outcome. In a second step, we further prune mention-pairs based on *value difference* and *unit agreement*. Finally, we sort mention-pairs according to classifier scores, and select top-k mention-pairs for each quantity mention based on *mention type* and *score distribution*.

A. Text-Mention Tagger

We tag text mentions, based on local features, with one of the following labels: *difference*, *sum*, *change ratio*, *percentage*, or *single cell*. Each of the four aggregation labels is associated with a small list of manually compiled cue words, such as "total, summed, overall, together" for *sum*, and analogous lists for the other tags. Likewise, words like "around, about, ca., approximately, nearly, almost" are considered as indicators for mention values being approximate. Observing the presence of such cue words in the proximity of a text mention is used for the following features that the tagger considers:

- *Approximation Indicator:* A categorical feature that specifies an approximation indicator accompanying the mention. The indicator is inferred from the immediate context of the text mention, where the immediate context is a window of 10 words around the text mention. The approximation indicator can take one of the following values: *approximate*, *exact*, *upper bound*, *lower bound*, and *none*.
- *Aggregation Function Features:* For each aggregation function we compute the count of supporting cue words in the mention context under the following scopes:
  1) *Immediate Context:* contains the tokens occurring within a window of 10 words around the text mention.
  2) *Local Context:* contains the tokens occurring in the same sentence with the text mention.
  3) *Global Context:* contains the tokens occurring in the same paragraph with the text mention.

- *Scale:* numerical value indicating the order of magnitude of the text mention.
- *Precision:* numerical value indicating the number of digits after the decimal point.
- *Unit:* a categorical feature that specifies the unit associated with the mention. The following is the list of units we consider: dollar, euro, percent, pound, and unknown unit.
- *Exact Match in Table(s):* the number of table mentions that exactly matches the surface form of the text mention. This number is summed up over all tables associated with the document.

We train the tagger, as a simple classifier, with a small labeled dataset, withheld from all other components and experiments. The tagger achieves high precision for the four kinds of aggregation functions. We intentionally optimize for high precision, at the expense of lower recall: the tagger sometimes confuses text mentions that match single cells with aggregates, incorrectly tagging them as *sum* or *diff* etc. However, this is not a problem as we can prune mention-pairs conservatively, by avoiding to eliminate single-cell matches at this stage. We use the tagger for the following pruning heuristics for mention-pairs:

- We keep all mention-pairs for single-cell mentions in tables.
- We prune aggregate mention-pairs if the aggregation function for the virtual cell does not match the predicted tag.

So this pruning step typically discards mention-pairs for all but one aggregation-function virtual cell, but keeps all mention-pairs with single cells. Further pruning steps for the single-cell cases are presented next.

B. Mention-Pair Pruning

Pruning based on Value Difference and Unit Mismatch: Based on the confidence scores returned by the mention-pair classifier, we prune mention-pairs whose numeric values differ by more than a threshold v if the classifier score is less than p. We tune the values of v and p on the withheld validation dataset. In addition, for mentions with specified units, we prune mention-pairs that disagree in unit.

After these pruning steps, we select the top-k candidate pairs for each text mention by the following criteria:

- **Mention Type:** We determine the mention type based on its surface form, context and the table mentions it potentially pairs with. A text mention can be *exact* (12.374), *approximate* (12.4) or *truncated* (12.3). First we rely on the context to determine the type of the quantity mention, by extracting quantity modifiers, such as 'approximately', 'exactly', and 'about'. If the context is insufficient to determine the mention type, we compare the surface form of the mention to that of potential table mentions with high confidence returned by the classifier. Then, we determine the mention type by majority vote. For example, if most of the high-confidence potential table mentions exactly match the text mention, then the text mention is exact. For exact mentions we pick the top k_{exact} mention-pairs and for approximate and truncated mentions we pick the top k_{approx} mention-pairs, where k_{exact} and k_{approx} are tunable parameters.
- **Distribution Entropy:** We consider the distribution of confidence scores returned by the classifier for the pairs with the same text mention. Sometimes, this distribution can be so skewed that only few candidates need to be kept, whereas in other cases a large number of candidates could be near-ties and should all be kept. To reflect this intuition, we compute the entropy of the distribution, and adjust k for the top-k candidates in proportion to the entropy. We set a specific threshold for the entropy value, and for distributions with entropy falling below this threshold, we pick the top k_s mention-pairs, otherwise we pick the top k_l mention-pairs, where k_s and k_l are tunable parameters.

Sales were up 5% on both a reported and organic basis, compared with the second quarter of 2012. Segment profit was up 11% and segment margins increased 60 bps to 13.3% primarily driven by strong productivity and volume leverage.

Table 1: Transportation Systems

($ Millions)	2Q 2012	2Q 2013	% Change
Sales	900	947	5%
Segment Profit	114	126	11%
Segment Margin	12.7%	13.3%	60 bps

Table 2: Automation & Control

($ Millions)	2Q 2012	2Q 2013	% Change
Sales	3,962	4,065	3%
Segment Profit	525	585	11%
Segment Margin	13.3%	14.4%	110 bps

Fig. 3. Example with Coupled Quantities

VI. Global Resolution

The need for joint inference over candidate pairs for multiple text mentions arises due to dependencies among mentions, which need to be harnessed to resolve ambiguities. Consider the example in Figure 3. The text mentions "11%" and "13.3%" have exact matches in both of the shown tables, and local-resolution algorithms cannot infer the proper alignment. However, when considering these two mentions jointly with "60 bps" and "5%", it becomes clear that all of these refer to the first table.

We have devised an unsupervised algorithm for this kind of global resolution. The algorithm encodes dependencies among mentions into a graph and uses random walks to infer the best joint alignment. We also considered an alternative algorithm based on constraint reasoning with Integer Linear Programming (ILP) and experimented with it, but that approach did not scale sufficiently well.

A human reader who glances a text mention and wants to identify to which table cell it refers, would first consider some matching values, including approximate or aggregate matches. These are candidate pairs, which we encode as edges in a graph, using the classifier's confidence scores as prior

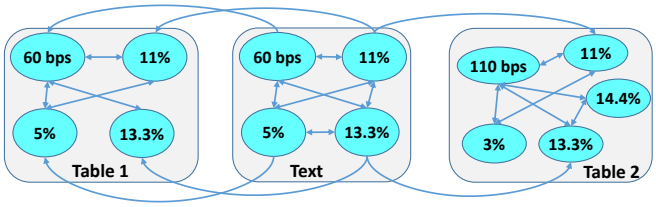

Fig. 4. Fragment of the Graph for Figure 3

edge weights. In case of ambiguity, the human user would then spot neighboring quantities in either text or table to assess the possible options and refine the hypothesis space of viable pairs. This would include looking at other quantities in textual proximity as well as other table cells in the same row or column. This intuition of human inference is cast into dependency edges between such context-related mentions, in both text and table—with weights based on relatedness strengths. Finally, the "strongest paths" connecting a text mention with table mention candidates determine the best alignment. We cast this intuition into a random walk over the weighted graph.

A. Graph Construction

We construct an undirected edge-weighted graph $G = (V, E)$ for each document:

- The node set V consists of all quantity mentions in the document's text and tables.
- The edge set E consists of three kinds of edges connecting related nodes: text-text edges, table-table edges, and text-table edges as explained below.

Text-text edges: There is an edge for each pair of text quantity mentions that are within a certain proximity or have similar surface forms. Edge weights (W_{xx}) are computed based on the following linear combination of proximity and string similarity:

$$W_{xx}(x_1, x_2) = \lambda_1 \mathrm{f}_{\mathrm{prox}}(x_1, x_2) + \lambda_2 \mathrm{f}_{\mathrm{strsim}}(x_1, x_2).$$

The hyperparameters λ_1 and λ_2 are tuned using grid search on the withheld validation dataset. We define $\mathrm{f}_{\mathrm{prox}}(x_1, x_2)$ as the number of tokens separating the two mentions, divided by the length of the document. String similarity $\mathrm{f}_{\mathrm{strsim}}(x_1, x_2)$ is defined as the Jaro-Winkler distance as described in Section IV-B.

Table-table edges: There is an edge for each pair of table quantity mentions in the same row or the same column of the same table. Edge weights W_{tt} are set uniformly for each pair of table mentions sharing the same row or the same column.

Text-table edges: There is an edge for each pair of text and table mention that is kept by the adaptive filtering stage. Edge weights (W_{xt}) are set to the confidence scores returned by the classifier. This can be viewed as an informed prior for the global resolution stage.

After this initial graph construction, all edge weights are normalized to obtain a stochastic graph, via dividing each node's outgoing weights by the total weight of these edges.

B. Graph Algorithm

Random walk with restart (RWR): Random walks have been widely used for ranking and alignment tasks over graphs (e.g. [15], [26], [34]), the most famous case being PageRank. In our setting, we employ random walks with restart: starting from a text mention, the graph is stochastically traversed, with a certain probability of jumping back to the initial node. This technique is also known as topic-specific or personalized PageRank [12]. It approximates the *stationary visiting probabilities* $\pi(t|x)$ of table-mention node t for walks starting from a text mention x. Our implementation iterates RWRs for each text mention until the estimated visiting probabilities of the candidate table mentions change by less than a specified convergence bound. This way we can rank the candidate table mentions t for the text mention x. Finally, this information is combined with the prior scores $\sigma(t|x)$ of the previous-stage classifier, leading to the overall scoring:

$$\mathrm{OverallScore}(t|x) = \alpha \cdot \pi(t|x) + \beta \cdot \sigma(t|x), \qquad (1)$$

with hyper-parameters α and β (which are tuned on the validation data).

Alignment decisions: The RWR from text mention x computes $\pi(t|x)$ for each table mention t. Pair $\langle x, t^* \rangle$ forms an alignment if and only if (i) t^* is the table mention with the highest overall score $\mathrm{OverallScore}(t^*|x)$, and (ii) its overall score $\mathrm{OverallScore}(t^*|x)$ exceeds a tunable threshold ϵ. Interestingly, making an alignment decision adds knowledge, and we propose to exploit that by updating the graph. In particular, after identifying an alignment $\langle x, t^* \rangle$, x cannot have alignments with any other table mention, and hence we modify the graph by removing all edges (x, t) for any $t \neq t^*$ (if no alignment is found for x, then all text-table edges adjacent to x are removed.) This way the next RWR for another text mention is able to leverage the new alignment information for improved results. This introduces a new issue: the order in which text mentions are processed. We discuss our approach to this next.

Entropy-based ordering: Note that a correct alignment decision will improve knowledge for future RWR executions, but an incorrect alignment decision can be harmful. Hence one intuitively should make decisions for the easier text mentions first, and then factor this information into the later decisions on the harder cases. To quantify the difficulty of aligning a text mention, we use the entropy of the classifier's confidence scores (see Section V). High entropy, close to uniform scores, means that there are several candidates among the table mentions that are not easy to distinguish. Low entropy, with highly skewed scores, indicates that there is one strongly preferred candidate—with the extreme case of having exactly one candidate only. Thus we process text mentions in order of

Algorithm 1: Graph-based global resolution

Data: undirected edge-weighted graph $G = (V, E)$; set C of mention-pair candidates $(x, t) \in E$ with prior confidence scores $\sigma(x, t)$

Result: subset $A \subseteq C$ of pairs for final alignment

$A := \emptyset$;
for *each text mention* x *with* $\exists t : (x, t) \in C$ **do**
  normalize $\{\sigma(x, t) : (x, t) \in C\}$ to a probability distribution;
  compute its entropy H(x);
end
for *each* x *in increasing order of H(x)* **do**
  run RWR from x to compute stationary probabilities $\pi(t|x)$ for all t with $(x, t) \in C$;
  $\text{OverallScore}(t|x) = \alpha \cdot \pi(t|x) + \beta \cdot \sigma(t|x)$;
  let $t^* := argmax_t \, \text{OverallScore}(t|x)$;
  if $\text{OverallScore}(t^*|x) > \epsilon$ **then**
    add (x, t^*) to A;
    delete edges (x, t) for all $t \neq t^*$ from G;
  end
  else
    delete edges (x, t) for all t from G;
  end
end

increasing entropy. Once an alignment is resolved for a text mention, only this text-table edge is kept and all edges to other table-mention candidates are removed.

Pseudo-code for the overall graph algorithm is given in Algorithm 1.

VII. Experimental Setup

A. Data

To evaluate BriQ, we use the Dresden Web Table Corpus (DWTC) which comprises about 125 Million tables extracted from 3.6 Billion web pages in the Common Crawl of July 2014 [8]. We compiled two datasets:

- **tableS**: a small annotated corpus from 495 web pages with complete assessment of ground-truth alignments, used to evaluate precision and recall of our method, and
- **tableL**: a large set from 1.5 million web pages, used to perform run-time measurements and demonstrate scalability of our method.

To construct the larger tableL corpus, we filtered the DWTC collection for web pages that meet a variety of criteria: English language, table(s) containing numerical cells, numerical mentions in text, overlap of tokens between table(s) and text. The resulting 1.5 million pages mostly fall under five major topics: finance, environment, health, politics, and sports (as determined by simple surface cues, and validated by manual sampling).

The tableS corpus is constructed from tableL by randomly selecting 505 pages and having them manually annotated by 8 hired annotators, all being non-CS students. We refrained from using mturk-like crowdsourcing for this purpose, as the annotation required fairly sophisticated guidelines and very thorough inspection of web pages; crowd workers would be unlikely to meet this quality assurance. In total, the 8 annotators spent about 130 person-hours on judging text-table mention pairs, and classifying them by their type: exact-match with single cell, sum, average, percentage, difference, ratio, minimum, maximum, unrelated, or other.

The inter-annotator agreement, with Fleiss' Kappa [9] being 0.6854, was substantial. All mention pairs confirmed by at least two annotators were kept, resulting in a final tableS corpus of 495 pages corresponding to 1,598 documents with 1,703 tables and 7,468 distinct text mentions of quantities.

B. Classifier Training

The tableS dataset was randomly split into disjoint training (80%), test (10%) and validation sets (10%). For each ground-truth mention pair in the training data (serving as positive samples), we automatically generated 5 negative samples by picking the table cells with the highest similarity to the positive sample (i.e., approximately the same values and similar context). These included many virtual cells for aggregate values, making the task very challenging. Table I gives a break-down of positive and negative samples by mention type.

We counter the label imbalance (#pos $\ll$ #neg) by giving different weights to the positive and negative labels in the classifiers' loss functions [14], [16]. These weights are inversely proportional to the ratio of the positive or negative labels in the dataset. The loss function is optimized for the area under the ROC curve, to ensure that neither precision nor recall could be neglected.

TABLE I
Classifier training data.

type	#pos	type	#neg
single-cell	4376	single-cell	3315
sum	267	sum	9300
percent	115	percent	4995
diff.	134	diff.	7924
ratio	141	ratio	5002
total	5039	total	39767

C. Metrics, Tuning and Testing

The traditional classifier performance metrics like accuracy and error rate are not informative in our setting with high imbalance between the positive and negative class. Therefore, we use precision, recall and F1 as major metrics to evaluate the BriQ system.

For tuning hyper-parameters, we use the withheld validation set of the annotated tableS corpus (10%). We use grid search to choose the best values for the hyper-parameters, for the classifiers as well as for the graph-based algorithm.

For testing classifiers, we use the withheld part of the annotated tableS corpus (10%). We apply the learned models

on all possible mention pairs between text and table (i.e., not just limited to the negative samples generated for training.) Overall, the test set has 687,321 mention pairs out of which only 0.1% are correct. The global resolution algorithm is tested with the outputs of the classifier and the adaptive filtering stage, typically reducing the size by two orders of magnitude.

D. Baselines

We compare BriQ against the following two baselines:

- *Classifier-only (RF)*: the Random Forest algorithm deployed in the first stage of BriQ, trained the same way as BriQ. For each text mention, the cell of the classifier's top-ranked mention-pair is chosen as output.
- *Random-Walk-only (RWR)*: a graph-based algorithm similar to the one used in the second stage of BriQ. The algorithm uses all features that are available to BriQ (see Section IV-B). However, as there are no prior probabilities computed from the first stage, these features are combined using uniform weights and then normalized to graph-traversal probabilities. Also, there is no pruning of any mention-pairs, making this baseline fairly expensive while still being an interesting comparison point.

We also considered an additional baseline derived from our earlier work on linking quantities to a knowledge base (QKB) [13]. Given a candidate mention-pair, we map both the text mention and the table cell to the QKB, this way normalizing them. Then we compare the two mentions if they are the same (i.e., link to the same QKB entry with exact-matching values). While this takes care of unit matching, it is limited to the units registered in the QKB and does not nearly cover all the diverse units in our large-scale input data. Moreover, the test can work only if the values of the two normalized mentions match exactly. For approximate matches where one text mention could be mapped to different single or virtual cells, the approach is unsuitable. Since approximate matches are very frequent in our test data, we did not pursue this possible baseline any further.

VIII. Experimental Results

A. Alignment Quality

We conducted experiments with three variations of text mentions, with increasing difficulty:

- **Original text mentions,** as given in the document. This is the main experiment.
- **Truncated text mentions,** where we removed the least significant digit of each original text mention. For example, 6746, 2.74, 0.19 became 6740, 2.7, and 0.1. This is meant as an additional test of robustness, making all test cases more difficult.
- **Rounded text mentions,** where we numerically rounded the least significant digit of each text mention. For example, 6746, 2.74, 0.19 became 6750, 2.7, and 0.2. This is meant as a stress test, with the additional challenge of making surface-form similarity less informative.

Original mentions: Table II shows the results for the original, truncated and rounded mentions. For the original mentions, BriQ outperforms both baselines, RF and RWR, by a large margin, regarding both precision and recall. BriQ achieved an F1 score of more than 70%, which is remarkably high given the noisy nature of the real-life data and the difficulty of the alignment problem.

TABLE II
Results for *original, truncated and rounded* text mentions.

	Original			Truncated			Rounded		
	RF	RWR	BriQ	RF	RWR	BriQ	RF	RWR	BriQ
recall	0.43	0.52	0.68	0.27	0.42	0.58	0.13	0.34	0.49
prec.	0.37	0.53	0.79	0.25	0.44	0.63	0.10	0.35	0.52
F1	0.40	0.53	0.73	0.26	0.43	0.60	0.11	0.34	0.51

Truncated and rounded mentions: As expected, the results for truncated and rounded mentions in Table II show a drop in quality, and the decrease is more pronounced for rounded mentions. In both of these situations, BriQ has the best results. For truncated mentions, BriQ still achieves fairly good quality, with an F1 score of ca. 60%. For rounded mentions, it achieves decent quality, with an F1 score of ca. 51%. In contrast, the two baselines degrade strongly. Especially, the RF classifier alone is not competitive at all, demonstrating our insight that the quantity alignment problem cannot be solved solely by supervised end-to-end machine learning.

Results by Mention Type: Tables III, IV and V break down the results by aggregation type: sum, difference, percentage, change ratio and single-cell match. BriQ clearly outperforms RF and RWR on all mention types and RWR outperforms RF on all types except for single-cell. As expected, BriQ has the best F1 score, 79%, on text mentions that refer to a *single* table cell. For sum and difference, BriQ achieved fairly good F1 scores of 72% and 43%, respectively. For the remaining two cases—percentage and ratio—all methods dropped substantially in output quality. The reason is that these cases are rather infrequent, so that the classifier gave them very low prior scores, a bias effect that the global resolution could not fully compensate.

TABLE III
Results by mention type for original mentions, using RF.

	sum	diff.	percent	change ratio	single-cell
recall	0.00	0.27	0.03	0.06	0.48
prec.	0.00	0.04	0.02	0.01	0.70
F1	0.00	0.06	0.03	0.02	0.57

TABLE IV
Results by mention type for original mentions, using RWR.

	sum	diff.	percent	change ratio	single-cell
recall	0.61	0.33	0.09	0.18	0.57
prec.	0.52	0.22	0.43	0.27	0.57
F1	0.56	0.26	0.15	0.21	0.57

Effectiveness of Adaptive Filtering: The adaptive filtering is crucial for BriQ to reduce the input size of the global resolution stage. Table VI shows the selectivity of our filters (i.e., the ratio of retained mention pairs to all mention pairs

TABLE V
RESULTS BY MENTION TYPE FOR ORIGINAL MENTIONS, USING BRIQ.

	sum	diff.	percent	change ratio	single-cell
recall	0.74	0.62	0.10	0.20	0.75
prec.	0.71	0.33	0.75	0.30	0.84
F1	0.72	0.43	0.17	0.24	0.79

that the classifier dealt with) and the recall after the filters. These numbers clearly demonstrate the enormous gains of the filtering stage. Conversely, the near-optimal recall numbers in the table show that we rarely make false-negative errors: BriQ effectively avoids erroneously dismissing good candidates from the mention-pair space.

TABLE VI
SELECTIVITY AND RECALL AFTER FILTERING.

type	selectivity	recall
sum	0.01	1.00
difference	0.01	0.87
percentage	< 0.01	0.91
change ratio	< 0.01	0.88
single-cell	0.04	0.91
overall	0.01	0.91

B. Ablation Study

We studied the influence of different feature groups on the two baselines and BriQ. We divide our feature space into three feature groups:

- **surface form similarity**.
- **context features**, including local and global word overlap, local and global noun phrases overlap, aggregate function match, and approximate indicator.
- **quantity features**, including relative value difference, unnormalized value difference, unit match, precision difference, and scale difference.

For the ablation study, we carried out three experiments, each corresponding to one feature group left out, thus training, tuning and testing the three models end-to-end on the remaining features. Table VII shows the F1 score, precision and recall of the three experiments in comparison with the full-feature model. The results underline the robustness of BriQ in comparison to the other baselines. Although BriQ's recall is affected by leaving out some features, its precision is stable. Leaving out context features leads to the highest degradation in BriQ's performance. Interestingly, leaving out the quantity features resulted in improvements of the RF classifier. The reason is that, without these features, the classifier has fewer virtual cells to consider (i.e., approximately matching values from aggregation of several table cells), making it easier to get the frequent single-cell cases right. However, BriQ still outperformed the RF classifier by a large margin.

C. Run-Time Results

BriQ is implemented in PySpark using Python, NetworkX, and SciPy libraries for the graph algorithm. For the RF classifier, we use R with the caret package, integrated into BriQ by the rpy2 library. All experiments were run on a Spark cluster with 10 executors, each with 6 cores and 30GB of memory, and with 50GB of driver memory. Training and tuning takes about 10 hours (on a very large dataset), with the grid search for the best hyper-parameters being the major factor (as it is often the case in machine learning). This is a *one-time* pre-processing effort.

TABLE VII
ABLATION STUDY: RECALL, PRECISION AND F1 SCORE

	Recall			Precision			F1		
	RF	RWR	BriQ	RF	RWR	BriQ	RF	RWR	BriQ
all features	0.43	0.52	0.68	0.37	0.53	0.79	0.40	0.53	0.73
w/o surf. sim.	0.37	0.36	0.65	0.33	0.39	0.77	0.35	0.37	0.70
w/o context	0.43	0.38	0.59	0.34	0.44	0.77	0.38	0.41	0.67
w/o quantity	0.43	0.31	0.61	0.54	0.35	0.77	0.48	0.33	0.68

To measure the run-time performance of BriQ for processing documents, we use the *tableL* dataset of about 1.5 million web pages. Table VIII shows the throughput of BriQ in terms of completed documents per minute, broken down into different thematic domains (e.g., quantities in finance are different in nature from quantities in sports). The throughput numbers clearly indicate that BriQ is practically viable at large scale. Moreover, it is 30 time faster than the RWR baseline that has a throughput of 76 documents per minute.

TABLE VIII
BRIQ THROUGHPUT BY DOMAIN.

	pages	documents	mentions	#docs/min
environment	118,724	986,180	3,062,943	2,935
finance	325,853	3,374,175	10,596,979	5,029
health	102,132	879,388	1,930,975	4,604
politics	128,318	2,762,873	4,123,800	6,223
sports	527,263	2,173,832	7,393,225	863
others	309,292	3,141,865	6,796,835	2,588
total	1,511,582	13,318,313	33,904,757	2,478

Table IX gives more statistics for each of these domains. We see that documents on sports led to a large number of virtual cells for aggregated values, incurring higher load and hence resulting in lower throughput than for the other domains.

TABLE IX
TABLE STATISTICS BY DOMAIN.

	rows	columns	single cells	virtual cells
environment	7	4	21	243
finance	7	4	16	142
health	3	2	4	26
politics	8	3	17	137
sports	8	6	35	523
others	7	4	21	252
average	7	4	19	220

IX. DISCUSSION

Anecdotal examples: Figure 5 shows three alignments computed by BriQ. Examples (a) and (b) illustrate the ability to detect and align change rates and percentages to the correct cell pairs. In example (c), BriQ is even able to discover the

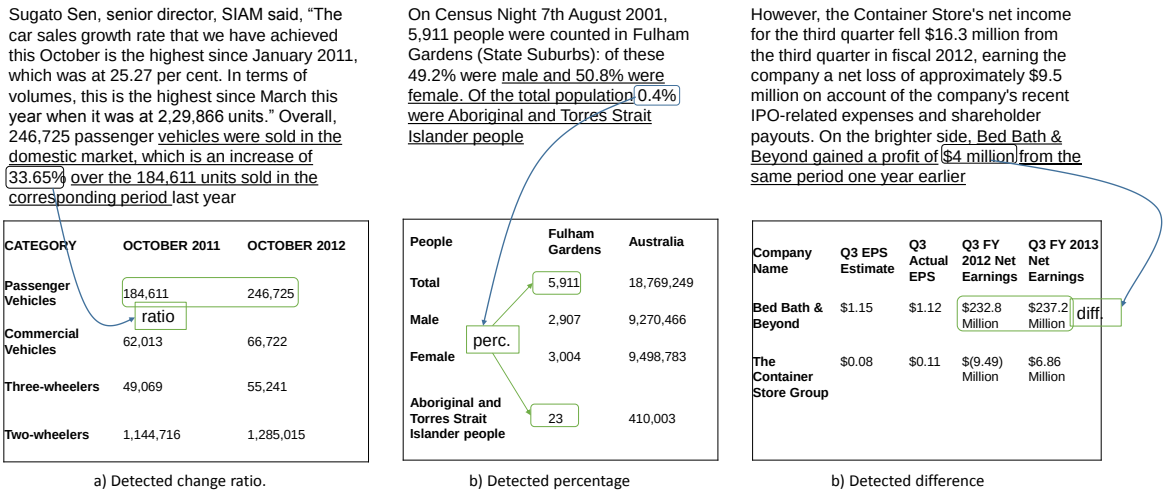

Fig. 5. Examples of alignments discovered by BriQ

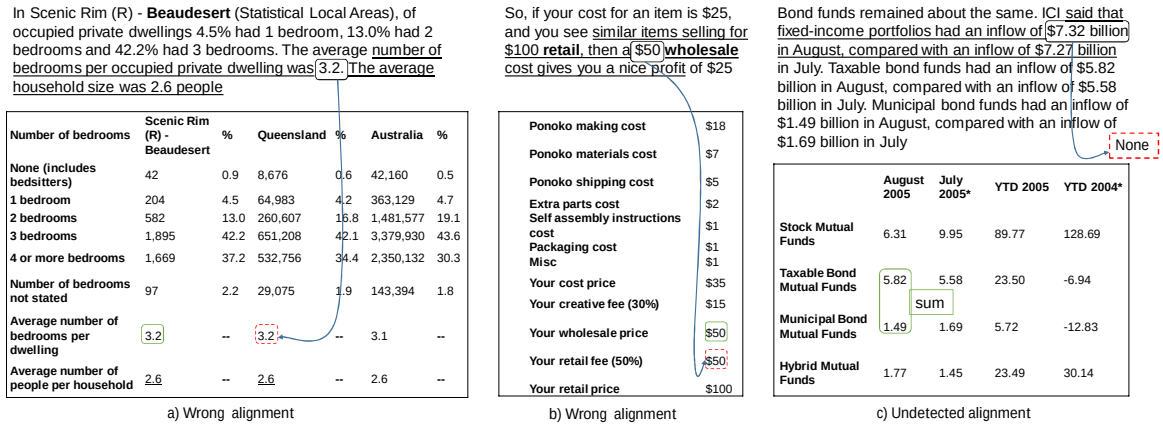

Fig. 6. Examples of errors made by BriQ

approximate difference between two cells and align it properly.

Typical error cases: Figure 6 shows some of the typical errors made by BriQ. The first case is in examples (a) and (b), having same-value collisions with several cells in the tables. In (a) the value ‘3.2’ exists in two cells in the same row with very similar context. As the immediate context of the quantity ‘3.2’ in the text, underlined, does not contain any words related to the columns, BriQ fails to identify the correct alignment. In (b) the immediate context of the quantity ‘$50’ contains both words ‘wholesale’ and ‘retail’. Moreover, the quantity ‘$100’ is closer to the incorrectly aligned cell ‘$50’. So BriQ fails here because of high ambiguity.

The third example (c) illustrates the case where the immediate context of the text mention ‘$7.32 billion’ has a single-word overlap with the table context, “August”. In addition the scale of the quantity (i.e., billion) is missing in the table. Such cases are extremely difficult to deal with, since neither the quantity features nor context features can help in finding the correct alignment.

X. Related Work

Web Tables: Schema-less ad-hoc tables embedded in Web pages have first been brought to the database research agenda by the seminal work of [3], [4], [21]. The focus of this work was on enabling search engines to include tables as results of keyword queries. Follow-up work tackled various forms of light-weight data integration, like matching names in table headers against queries, matching attributes of different tables with each other, and inferring approximate schemas (e.g., [18], [19], [27], [35]).

Entity Linking: Mapping names of people, places, products, etc. onto canonicalized entities in a knowledge base has received great attention in the literature; a recent survey is given by [33]. This work has mostly focused on surface names in text documents. The most notable exceptions that addressed names in tables (in combination with mapping column headers) are [2], [19], [28]. Their methods for entity linking vary from context-similarity-based rankings and simple classifiers to advanced forms of probabilistic graphical models for joint inference over a set of mentions.

Quantity Extraction: Recent work has addressed the task of recognizing quantities in text and extracting them as proper mentions (including units, reference range, etc.) [1], [13], [20], [23], [29]–[32]. These methods are based on pattern matching and/or machine learning models like Conditional Random Fields. However, only [13], [31] go beyond mere extraction and aim to canonicalize quantity mentions by linking them to a knowledge base of measures and units. In doing this, they rely on an explicit – in their cases small and manually crafted – knowledge base, though. This approach is limited in scope and does not scale to the wide diversity of quantities in large collections of Web tables. The BriQ approach, on the other hand, does not require an explicit knowledge base and copes with the full scope of possible inputs.

Coreference Resolution in NLP: A very different domain with resemblance to our problem of quantity alignment is the task of coreference resolution in natural language processing (NLP). Given a text document with entity names as well as underspecified expressions like pronouns ("he", "she", "her" etc.) and common noun phrases (e.g., "the lead singer", "the founder of Apple" etc.), the task is to compute equivalence classes of coreferences. For example, pronouns should be linked to a name in the same or a preceding sentence. State-of-the-art methods for this problem are mostly based on rules and/or machine-learning techniques for clustering or classification (e.g., [6], [7], [11], [17], [24]). None of these considers mentions of quantities, though.

XI. Conclusion

We have introduced the new problem of aligning quantities between text and tables. Our methodology combines supervised classification based on local contexts, adaptive filtering techniques for computational tractability, and joint inference methods for global resolution. Comprehensive experiments with ad-hoc web tables show that all stages of this pipeline are essential, and together can achieve good precision and recall at affordable computational cost.

As for future work, we plan to investigate this problem also in the context of enterprise content (e.g., spreadsheets in documents) and specialized domains such as material science or biomedical documents. Quantity alignment is an important step towards semantically understanding numbers in unstructured and semi-structured content. This in turn can open up the path towards next-generation search engines that can handle queries about quantities, such as Internet companies with annual income above 5 Mio. USD, electric cars with energy consumption below 100 MPGe (or equivalently, ca. 21 kWh/100km), or clinical trials with a daily anti-coagulant dosage above 30 mg. All these examples are way beyond the scope of today's search engines; quantity understanding would bring them closer to feasibility.

References

[1] O. Alonso and T. Sellam. Quantitative information extraction from social data. *SIGIR*, 2018.

[2] C. S. Bhagavatula, T. Noraset, and D. Downey. TabEL: Entity linking in web tables. *ISWC*, 2015.

[3] M. J. Cafarella, A. Halevy, and J. Madhavan. Structured data on the web. *CACM*, Feb. 2011.

[4] M. J. Cafarella, A. Halevy, D. Z. Wang, E. Wu, and Y. Zhang. Webtables: exploring the power of tables on the web. *VLDB*, 2008.

[5] R. Caruana and A. Niculescu-Mizil. An empirical comparison of supervised learning algorithms. *ICML*, 2006.

[6] K. Clark and C. D. Manning. Improving coreference resolution by learning entity-level distributed representations. *ACL*, 2016.

[7] G. Durrett and D. Klein. A joint model for entity analysis: Coreference, typing, and linking. *TACL*, 2014.

[8] J. Eberius, M. Thiele, K. Braunschweig, and W. Lehner. Top-k entity augmentation using consistent set covering. *SSDBM*, 2015.

[9] J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 1971.

[10] M. Gambhir and V. Gupta. Recent automatic text summarization techniques: a survey. *Artificial Intelligence Review*, 47(1):1–66, 2017.

[11] A. Haghighi and D. Klein. Simple coreference resolution with rich syntactic and semantic features. *EMNLP*, 2009.

[12] T. H. Haveliwala. Topic-sensitive pagerank: A context-sensitive ranking algorithm for web search. *TKDE*, 2003.

[13] Y. Ibrahim, M. Riedewald, and G. Weikum. Making sense of entities and quantities in web tables. *CIKM*, 2016.

[14] N. Japkowicz and S. Stephen. The class imbalance problem: A systematic study. *Intelligent data analysis*, 6(5):429–449, 2002.

[15] N. Lao, T. M. Mitchell, and W. W. Cohen. Random walk inference and learning in A large scale knowledge base. *EMNLP*, 2011.

[16] S. Lawrence, et al. Neural network classification and prior class probabilities. In *Neural networks: tricks of the trade*. Springer, 1998.

[17] H. Lee, et al. Deterministic coreference resolution based on entity-centric, precision-ranked rules. *COLING*, 2013.

[18] O. Lehmberg and C. Bizer. Stitching web tables for improving matching quality. *PVLDB*, 2017.

[19] G. Limaye, S. Sarawagi, and S. Chakrabarti. Annotating and searching web tables using entities, types and relationships. *VLDB*, 2010.

[20] A. Madaan, A. Mittal, Mausam, G. Ramakrishnan, S. Sarawagi. Numerical relation extraction with minimal supervision. *AAAI*, 2016.

[21] J. Madhavan, L. Afanasiev, L. Antova, and A. Y. Halevy. Harnessing the deep web: Present and future. *CIDR*, 2009.

[22] A. Nenkova and K. McKeown. A survey of text summarization techniques. *Mining text data*, 2012.

[23] S. Neumaier, J. Umbrich, J. X. Parreira, and A. Polleres. Multi-level semantic labelling of numerical values. *ISWC*, 2016.

[24] V. Ng. Machine learning for entity coreference resolution: A retrospective look at two decades of research. *AAAI*, 2017.

[25] A. Niculescu-Mizil and R. Caruana. Predicting good probabilities with supervised learning. *ICML*, 2005.

[26] M. T. Pilehvar, D. Jurgens, and R. Navigli. Align, disambiguate and walk: A unified approach for measuring semantic similarity. *ACL*, 2013.

[27] R. Pimplikar and S. Sarawagi. Answering table queries on the web using column keywords. *VLDB*, 2012.

[28] D. Ritze, O. Lehmberg, and C. Bizer. Matching HTML tables to dbpedia. *WIMS*, 2015.

[29] S. Roy, T. Vieira, and D. Roth. Reasoning about quantities in natural language. *TACL*, 2015.

[30] S. Saha, H. Pal, and Mausam. Bootstrapping for numerical open IE. *ACL*, 2017.

[31] S. Sarawagi and S. Chakrabarti. Open-domain quantity queries on web tables: annotation, response, and consensus models. *SIGKDD*, 2014.

[32] T. Sellam and O. Alonso. Raimond: Quantitative data extraction from twitter to describe events. *ICWE*, 2015.

[33] W. Shen, J. Wang, and J. Han. Entity linking with a knowledge base: Issues, techniques, and solutions. *TKDE*, 2015.

[34] H. Tong, C. Faloutsos, and J. Pan. Fast random walk with restart and its applications. *ICDM*, 2006.

[35] P. Venetis, et al. Recovering semantics of tables on the web. *PVLDB*, 2011.