Entities with Quantities: Extraction, Search, and Ranking

Vinh Thinh Ho

Dissertation zur Erlangung des Grades
des DOKTORS DER INGENIEURWISSENSCHAFTEN (DR.-ING.)
der Fakultät für Mathematik und Informatik
der Universität des Saarlandes

Saarbrücken, 2022

Day of Colloquium	28/10/2022
Dean of the Faculty	Univ.-Prof. Dr. Jürgen Steimle
Chair of the Committee	Prof. Dr. Dietrich Klakow
Reporters	
First Reviewer	Prof. Dr. Gerhard Weikum
Second Reviewer	Dr. Daria Stepanova
Third Reviewer	Prof. Dr. Martin Theobald
Academic Assistant	Dr. Roohani Sharma

Abstract

Quantities are more than numeric values. They denote measures of the world's entities such as heights of buildings, running times of athletes, energy efficiency of car models or energy production of power plants, all expressed in numbers with associated units. Entity-centric search and question answering (QA) are well supported by modern search engines. However, they do not work well when the queries involve quantity filters, such as searching for athletes who ran 200m under 20 seconds or companies with quarterly revenue above $2 Billion. State-of-the-art systems fail to understand the quantities, including the condition (less than, above, etc.), the unit of interest (seconds, dollar, etc.), and the context of the quantity (200m race, quarterly revenue, etc.). QA systems based on structured knowledge bases (KBs) also fail as quantities are poorly covered by state-of-the-art KBs.

In this dissertation, we developed new methods to advance the state-of-the-art on quantity knowledge extraction and search. Our main contributions are the following:

- First, we present Qsearch [Ho et al., 2019, Ho et al., 2020] – a prototype system that can handle advanced queries with quantity constraints using cues present in both the query and the text sources. Qsearch has two main contributions. The first contribution is a deep neural network model specifically designed for extracting quantity-centric tuples from text sources. The second contribution is a novel query matching model to retrieve and rank the extracted tuples.

- Second, to tap into heterogeneous tables, we present QuTE [Ho et al., 2021a, Ho et al., 2021b] – a system for extracting quantity facts from web sources, specifically ad-hoc web tables in HTML pages. The contribution of QuTE includes a method for aligning quantity columns with entity columns, by leveraging evidence from external text corpora. For answering quantity queries, we contextualize the extracted entity-quantity pairs with informative cues from the table, and present a new method for corroborating and re-ranking candidate answers based on inter-fact consistency.

- Third, we present QL [Ho et al., 2022] – a recall-oriented method for enhancing knowledge bases with quantity facts. State-of-the-art KBs, such as Wikidata or YAGO, cover many entities and their relevant information, but often miss out on important quantitative properties. Our method QL is query-driven, and based on iterative learning, with two main contributions to enhance KB coverage. The first contribution is a query expansion technique to capture a larger pool of fact candidates. The second contribution is a technique for self-consistency from observations on quantity value distributions.

Kurzfassung

Zahlen sind mehr als nur numerische Werte. Sie beschreiben Maße von Entitäten wie die Höhe von Gebäuden, die Laufzeit von Sportlern, die Energieeffizienz von Automodellen oder die Energieerzeugung von Kraftwerken - jeweils ausgedrückt durch Zahlen mit zugehörigen Einheiten. Entitätszentriete Anfragen und direktes Question-Answering werden von Suchmaschinen häufig gut unterstützt. Sie funktionieren jedoch nicht gut, wenn die Fragen Zahlenfilter beinhalten, wie z. B. die Suche nach Sportlern, die 200m unter 20 Sekunden gelaufen sind, oder nach Unternehmen mit einem Quartalsumsatz von über 2 Milliarden US-Dollar. Selbst moderne Systeme schaffen es nicht, Quantitäten, einschließlich der genannten Bedingungen (weniger als, über, etc.), der Maßeinheiten (Sekunden, Dollar, etc.) und des Kontexts (200-Meter-Rennen, Quartalsumsatz usw.), zu verstehen. Auch QA-Systeme, die auf strukturierten Wissensbanken ("Knowledge Bases", KBs) aufgebaut sind, versagen, da quantitative Eigenschaften von modernen KBs kaum erfasst werden.

In dieser Dissertation werden neue Methoden entwickelt, um den Stand der Technik zur Wissensextraktion und -suche von Quantitäten voranzutreiben. Unsere Hauptbeiträge sind die folgenden:

- Zunächst präsentieren wir Qsearch [Ho et al., 2019, Ho et al., 2020] – ein System, das mit erweiterten Fragen mit Quantitätsfiltern umgehen kann, indem es Hinweise verwendet, die sowohl in der Frage als auch in den Textquellen vorhanden sind. Qsearch umfasst zwei Hauptbeiträge. Der erste Beitrag ist ein tiefes neuronales Netzwerkmodell, das für die Extraktion quantitätszentrierter Tupel aus Textquellen entwickelt wurde. Der zweite Beitrag ist ein neuartiges Query-Matching-Modell zum Finden und zur Reihung passender Tupel.

- Zweitens, um beim Vorgang heterogene Tabellen einzubinden, stellen wir QuTE [Ho et al., 2021a, Ho et al., 2021b] vor – ein System zum Extrahieren von Quantitätsinformationen aus Webquellen, insbesondere Ad-hoc Webtabellen in HTML-Seiten. Der Beitrag von QuTE umfasst eine Methode zur Verknüpfung von Quantitäts- und Entitätsspalten, für die externe Textquellen genutzt werden. Zur Beantwortung von

Fragen kontextualisieren wir die extrahierten Entitäts-Quantitäts-Paare mit informativen Hinweisen aus der Tabelle und stellen eine neue Methode zur Konsolidierung und verbesserteer Reihung von Antwortkandidaten durch Inter-Fakten-Konsistenz vor.

- Drittens stellen wir QL [Ho et al., 2022] vor – eine Recall-orientierte Methode zur Anreicherung von Knowledge Bases (KBs) mit quantitativen Fakten. Moderne KBs wie Wikidata oder YAGO decken viele Entitäten und ihre relevanten Informationen ab, übersehen aber oft wichtige quantitative Eigenschaften. QL ist frage-gesteuert und basiert auf iterativem Lernen mit zwei Hauptbeiträgen, um die KB-Abdeckung zu verbessern. Der erste Beitrag ist eine Methode zur Expansion von Fragen, um einen größeren Pool an Faktenkandidaten zu erfassen. Der zweite Beitrag ist eine Technik zur Selbstkonsistenz durch Berücksichtigung der Werteverteilungen von Quantitäten.

Acknowledgments

First of all, I would like to give my special thank to my supervisor Prof. Gerhard Weikum for his invaluable guidance throughout my doctoral studies. He not only gave me a plenty of ideas for my thesis, but also sent me so much energy, enthusiasm, freedom when doing research, and many advice for my future career. Without his help, I would not have been able to finish this dissertation.

I would like to say thank to Prof. Koninika Pal for being a great collaborator. During my PhD, we had a lot of discussions and I received a plenty of insightful comments from her about the work. Together, we successfully published papers in many top-tier conferences.

I am also thankful for the support from Dr. Daria Stepanova during my studies. Daria was my master thesis advisor, the person who has brought me to and ignited my interest in research. During my PhD, she has given me a great internship opportunity at Bosch AI, where I have gained so much experience in conducting industrial research.

I would like to say thank to Prof. Martin Theobald for being the additional reviewer and examiner of my dissertation, and many thanks to Prof. Dietrich Klakow and Dr. Roohani Sharma for being the chair and the academic assistant of my PhD defense.

I also would like to thank my co-authors Prof. Klaus Berberich, Dr. Simon Razniewski, Dr. Jannik Strötgen, Dr. Yusra Ibrahim and Dragan Milchevski for their great team work and insightful discussions. Thank Dr. Abdalghani Abujabal and Mohamed Soliman for their great guidance during my internship at Amazon. Many thanks to all my colleagues and staff members at D5 group.

I am also very grateful to uncle Duy Ta, aunt Hong Le, and my friends Canh Nguyen, Giang Nguyen for their help in my life. Many thanks to Hai Dang Tran, Xuan Cuong Chu, Ba Dat Nguyen, Tuan Phong Nguyen, Thac Thong Nguyen, Tien Thanh Cao, Soumen Ganguly, Noshaba Cheema and all my other friends for their support and advice during my PhD life.

Lastly but most importantly, I would like to send my sincere thank to my grandparents, my parents and my brother, who are always beside me, providing me with love and encouragement to continually push myself to success.

To my little family Bich Ngoc, Tue Linh and Ha Vy ♡

Contents

Chapter 1

Introduction

1.1 Motivation

The rapid expansion of the Internet has been accompanied by modern search engines such as Google Search, Microsoft Bing, Baidu, Yandex, and others. Traditionally, search engines handle user queries by returning ten blue links along with preview snippets, corresponding to the most relevant documents to the input queries. However, the user would still have to open and browse these result pages to find the desired information. The desired information may be anywhere on the web: in unstructured content like text in web pages, in semi-structured content like HTML tables, or in non-textual content like audio or video.

Entity-centric Search. Nowadays, with the emergence of large knowledge graphs (KG) and advances in natural language understanding (NLU), search engines have become more powerful. By detecting entity mentions in both user queries and page contents and linking them to knowledge graphs, search engines are able to answer entity-centric queries about people, places and products accurately and concisely. For example, Google can give direct answers for the following queries:

- **Query:** CEO of Amazon **Answer:** Andy Jassy
- **Query:** capital of Australia **Answer:** Canberra
- **Query:** Zuckerberg net worth **Answer:** 82.8 billion USD
- **Query:** CEOs of German companies **Answers:** Benno Dorer, Joe Kaeser, Dieter Zetsche, Christian Klein, ...

Search engines have gone a long way towards understanding entities and their types in online contents by learning information extraction and leveraging their knowledge graphs.

Entities with Quantities. Nonetheless, there is a dimension that is still largely overlooked and thus poorly supported: *quantities*. Quantities are more than mere numbers. They are relevant properties of entities, such as revenue of companies, energy efficiency of cars or distance and brightness of stars and galaxies. Quantities appear in many online sources: news articles, financial reports, scientific research results, medical records and more. Understanding them is a crucial step towards building next-generation information retrieval and question answering systems.

Quantities alone are meaningless, unless they are coupled with entities, in a proper context. Entity-centric search, with quantities being the target answer, is supported by modern search engines and QA systems, such as retrieving the net worth of Mark Zuckerberg or the height of the Eiffel Tower. Handling such look-up queries does not require any special treatment for the quantities, compared to other property look-ups, such as a person's nationality or a building's address. Quantities can be represented as *(measure, value, unit)* triples, such as *(height, 324, meter)*. The units can be simple such as meters, kilograms, Euros, etc., with well-defined conversion-rules between units of the same measure. But they can also be quite complex such as *km/h* or *m^3/s*, and even more sophisticated, especially for units from scientific domains such as *$Pa\text{-}s/m^3$* (for acoustic impedance), *$rad\text{-}m^2/mol$* (for molar optical rotatory power) or *W/mK* (for thermal conductivity) and more.

Quantity Queries. Entities with quantities brings the potential and the need for handling an interesting and challenging kind of queries, which is still underexplored and hardly supported by current search engines and QA systems: *searching for entities with quantity-filter conditions*. Consider the following example queries:

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

Depending on the values in these queries and the way the units are specified, major search engines occasionally return good results for some of the queries. For example, the third query, with the twist that the unit is changed to miles, returns a short list of car models ranked by electric range; however, all these are from a list of a single web page. Changing the condition of the query from 50 km to 45 km returns a different list where all results are incorrect (their electric range is lower than 45 miles). If we change the value to 57, the search engine falls back to the ten-blue-links mode and returns web pages rather than entities. The underlying reason for these deficiencies is that the search engines do not understand quantities. Handling this kind of queries requires

Entity Type/Property	**Wikidata**			**DBpedia**		
	#E	*#P*	*#Q*	*#E*	*#P*	*#Q*
car model/range	3195	4	4	6705	0	0
car model/engine power	3195	0	0	6705	0	0
mobile phone/display size	291	0	0	1358	1309	0
marathon runner/best time	1629	18	18	3426	1346	601

Table 1.1: Statistics on exemplary quantitative properties from Wikidata and DBpedia. *#E*: number of entities; *#P*: with property present; *#Q*: with explicit data type for the property.

interpreting quantities in terms of measure, value and unit, and reasoning to evaluate the filter conditions. Search engines and QA systems merely match tokens that denote numbers and units as if they were keywords. The occasional cases with good results are, by and large, accidental (drawing from many high-quality lists and tables within web pages).

Searching for entities with quantity conditions is important, especially for advanced users, such as analysts, journalists, scientists and other knowledge workers [Weikum, 2020]. For answering this kind of queries, one can, in principle, tap into structured data sources as a solution such as knowledge bases or open databases, accessible through web APIs. However, the relevant databases are scattered across the Internet, posing obstacles in discovering them and understanding their schemas, options for joins and unions, trustworthiness, etc. Knowledge bases, such as Wikidata, have good coverage of entities, but barely contain quantitative properties (e.g., 2000 marathon runners but only 20 with their best time captured, 3000 car models but none with engine power, energy efficiency, carbon footprint, etc.). Not just being limited, their quantitative facts are mostly literals, apart from dates, merely represented as strings; e.g., battery capacity of the BMW i3 is shown as the string *"i3 94 A·h: 33 kWh lithium-ion battery"* in DBpedia. Table 1.1 gives exemplary numbers for the quantity coverage in Wikidata and DBpedia. Consider even more complex example queries:

- Which is the most energy-efficient hybrid car model?
- How many female runners have completed a marathon race under 2:25h at least five times in their career?

These analytic queries raise issues about the completeness and freshness of the underlying data sources. Does the input contain all hybrid car models? Does the input contain all runners and all their races? Is the data up-to-date? An open knowledge base or database satisfying these desiderata does not exist. Only the web as a whole is the data source that has enough information needed to compute the accurate and complete answers for these queries.

This dissertation lays the first bricks in building a solution for answering quantity queries using data from the web.

1.2 Challenges and Scope

1.2.1 Challenges

Building a search system for answering quantity queries poses several challenges that need to be overcome:

Challenge C1: Quantity recognition and normalization. Quantities appear in very diverse and potentially noisy forms. Quantity values can be represented in different ways (e.g., “100000”, “100,000”, “100,000.00”, “one hundred thousand”), possibly with a scaling factor (e.g., “100k”, “100 thousand”) and an approximation indicator (e.g., “about”, “under”, “over”, “almost”). The same happens for units, even with the basic ones, (e.g., “metre”, “metres”, “meter”, “meters”, “m”). Capitalization also does matter (e.g., “MB” vs. “Mb” for megabyte and megabit). Combinations of quantity values and units in text are, hence, even harder to recognize (e.g., “100-meter”, “100m”, “100 metres” – with/without hyphens and spaces). A rule-based approach is often not sufficient for correctly detecting quantities, as disambiguation is usually needed. For example, token “m” might refer to either “meter” or “million”, token “k” might refer to “thousand” or “Kelvin”. Taking the quantity surrounding context into account is crucial for the correct recognition and interpretation of quantities.

Extracting quantities in semi-structured data such as web tables or lists even poses more challenges, e.g., quantities values and units appear in different table cells (header and body), quantity context scattered over different places: table cells, caption, page title, or surrounding text.

Challenge C2: Making sense of quantities. An essential step towards understanding quantities is to couple them with proper entities and context. Named entity recognition and disambiguation has been intensively studied in prior work. However, determining the entities to which the quantities refer, and finding the relation between them, are still big challenges. Very often, the linked entities do not appear in the same sentence as the quantities; instead their references are in the form of nominal and pronominal mentions. In web tables, entities and quantities are usually located in different cells.

Sometimes, entities do not appear inside the table, but in the table caption, the page title, the section titles, or being hidden in the table surrounding text.

Prior work on fact extraction usually resorted to pre-existing triples in a KB as the ground-truth to train a fact extractor through distant supervision. However, a richly populated knowledge base of quantity facts does not exist.

Challenge C3: Reasoning over quantities. Quantity filtering is not limited to just numeric comparisons. Even a basic filter can still present enormous challenges. Consider the example query "sprinters running 200m under 21s". An ideal retrieval system should be able to detect and disregard irrelevant information such as "Bolt has broken the world record with a time of 9.58s", by reasoning that 9.58s is not a reasonable run-time for a 200m race, even though the exact race-distance (100m) is not explicitly given.

The web contains all kinds of information, also including wrong and noisy cues (e.g., from untrustworthy sources, or just due to typing mistakes). Reasoning over quantities is an important step to recognize and exclude false candidates from the answers.

Challenge C4: Recall. A fundamental reason of directly tapping into the web for answering quantity queries is recall; otherwise we can just leverage some well-curated open database with a clear schema to retrieve few answers with very high precision. To this end, the capabilities of denoising, deduplicating, joining and aggregating over pieces of quantitative information are needed, especially for handling complex queries.

1.2.2 Scope

Addressing challenge C1, prior work has looked into recognizing and extracting numeric expressions from text using techniques like CRFs and LSTMs (e.g., [Alonso and Sellam, 2018, Roy et al., 2015, Saha et al., 2017]). Few works attempted to canonicalize quantities by mappings to hand-crafted knowledge bases of measures [Ibrahim et al., 2016]. For recognizing quantities on web tables, the most promising has been the work by Sarawagi et al. [Sarawagi and Chakrabarti, 2014], though with limited support for understanding measures and units.

In this dissertation, we focus on solving challenges C2, C3 and C4. Our methods build on state-of-the-art works for quantity recognition, and make contributions on making sense of quantities, reasoning over quantities, and tackling the recall problem.

1.3 Contributions

This dissertation has developed the following new methods to advance the state of the art:

Qsearch. We present Qsearch [Ho et al., 2019, Ho et al., 2020], a search and QA system, that can effectively answer advanced queries with quantity conditions. Our solution is based on a deep neural network for extracting quantity-centric tuples from text sources, and a novel matching model to retrieve and rank answers from news articles and other web pages. Experiments demonstrate the effectiveness of Qsearch on benchmark queries collected by crowdsourcing.

QuTE. We present QuTE [Ho et al., 2021a, Ho et al., 2021b], a novel method for automatically extracting quantity facts from ad-hoc web tables, leveraging data from text sources. 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.

QL. State-of-the-art knowledge bases (KBs), such as Wikidata, cover many relevant entities but often miss the corresponding quantities. Prior work on extracting quantity facts from web contents focused on high precision for top-ranked outputs, but did not tackle the KB coverage issue. We present QL [Ho et al., 2022], a recall-oriented approach which aims to close this gap in knowledge-base coverage. Our method is based on iterative learning for extracting quantity facts, with two novel contributions to boost recall for KB augmentation without sacrificing the quality standards of the knowledge base. The first contribution is a query expansion technique to capture a larger pool of fact candidates. The second contribution is a novel technique for harnessing observations on value distributions for self-consistency. Experiments with extractions from more than 13 million web documents demonstrate the benefits of our method.

1.4 Publications

Parts of this dissertation have been published in top-tier academic venues. Below is the list of published papers, where their first author is also the author of this dissertation.

- **Qsearch: Answering Quantity Queries from Text.** Vinh Thinh Ho, Yusra Ibrahim, Koninika Pal, Klaus Berberich, Gerhard Weikum. In Proceedings of *The 18th International Semantic Web Conference (ISWC)*, 2019.
- **Entities with Quantities: Extraction, Search, and Ranking.** Vinh Thinh Ho, Koninika Pal, Niko Kleer, Klaus Berberich, Gerhard Weikum. In Proceedings of *The 13th ACM International Conference on Web Search and Data Mining (WSDM)*, 2020.
- **Extracting Contextualized Quantity Facts from Web Tables.** Vinh Thinh Ho, Koninika Pal, Simon Razniewski, Klaus Berberich, Gerhard Weikum. In Proceedings of *The 30th Web Conference (WWW)*, 2021.
- **QuTE: Answering Quantity Queries from Web Tables.** Vinh Thinh Ho, Koninika Pal, Gerhard Weikum. In Proceedings of *The 2021 ACM Special Interest Group on Management of Data Conference (SIGMOD)*, 2021.
- **Enhancing Knowledge Bases with Quantity Facts.** Vinh Thinh Ho, Daria Stepanova, Dragan Milchevski, Jannik Strötgen, Gerhard Weikum. In Proceedings of *The 31st Web Conference (WWW)*, 2022.

1.5 Organization

The remainder of the dissertation is organized as follows. Chapter 2 gives background on methodology relevant for this dissertation. Chapter 3 and 4 present our methods for extracting quantity facts and answering quantity queries from two major types of web content, namely text and web tables. Chapter 5 presents our recall-oriented approach for enhancing KBs with quantity facts. Finally, Chapter 6 concludes the dissertation and outlines possible directions for future work.

Chapter 2

Background

2.1 Knowledge Graphs

Data on the web is represented in various forms: in unstructured content like text in web pages, in semi-structured content like HTML web tables, or in non-textual content like audio or visual. This poses obstacles for the computers, calling for methods for extracting, deduplicating, denoising, aggregating, analyzing and reasoning. With the advances of natural language understanding, computers can now accquire knowledge, by transforming the web content with high-heterogeneity into a structured form called knowledge graphs (KGs), or originally known as knowledge bases (KBs).

On the web, knowledge graphs are often encoded using the Resource Description Framework (RDF) data model [Lassila and Swick, 1999], as a collection of SPO triples ⟨*subject, predicate, object*⟩, reflecting facts about the world. The *subject* and *object* are the entities representing people, places, organizations, etc., and the *predicate* represents the relation between them. For example, knowledge about the Eiffel Tower might include the following facts:

- ⟨*Eiffel-Tower, location, Champ-de-Mars*⟩
- ⟨*Eiffel-Tower, named-after, Gustave-Eiffel*⟩
- ⟨*Eiffel-Tower, architect, Stephen-Sauvestre*⟩

The object of a KG fact can also be a literal – either a string, a numeric value, or a date, representing various properties of the entities such as height, mass, date of birth, etc., or other useful information such as a short description text or a website URL. For instance:

- ⟨*Eiffel-Tower, height, 324 metres*⟩
- ⟨*Eiffel-Tower, mass, 10100 tons*⟩
- ⟨*Eiffel-Tower, description, "tower located on the Champ de Mars in Paris, France"*⟩
- ⟨*Eiffel-Tower, official-website, "http://www.toureiffel.paris/"*⟩

Apart from capturing information about entities and their relations, knowledge graphs also include a typing system, containing facts about their entity types, through a special predicate *"type"*. For example:

- ⟨*Eiffel-Tower, type, lattice-tower*⟩
- ⟨*Eiffel-Tower, type, observation-tower*⟩
- ⟨*Eiffel-Tower, type, tourist-atraction*⟩

Entity types can also be subsets/supersets of each other, represented through the predicate *"subclass-of"*. For example:

- ⟨*lattice-tower, subclass-of, tower*⟩
- ⟨*tower, subclass-of, building*⟩

Entities with their types, together with the "subclass-of" relations between types, form a hierarchical taxonomy graph, which is valuable for many search, discovery and reasoning tasks over KGs.

2.1.1 State-of-the-art KGs

Early KGs have been crafted manually (e.g., Cyc [Lenat, 1995], WordNet [Miller, 1995]), resulting in high-quality KGs, but limited in their size and scope. Only until recently, with the expansion of the internet, large-scale manually-constructed KGs have come into existence. Prominent projects are Freebase [Bollacker et al., 2008], and Wikidata [Vrandecic and Krötzsch, 2014]. Some KGs have been constructed through semi(-automated) extraction techniques such as regular expressions, part-of-speech tags and other rules, to extract information from high-quality semi-structured data sources such as Wikipedia infoboxes and category pages. Prominent example KGs are YAGO [Suchanek et al., 2007], and DBpedia [Lehmann et al., 2015]. KGs are also constructed from unstructured textual sources such as KnowledgeVault [Dong et al., 2014] or NELL [Carlson et al., 2010], by leveraging advanced natural language processing and machine learning techniques for the automatic information extraction.

Being built by different communities, companies and organizations, knowledge graphs can also be domain-specific. They represent knowledge about commonsense (e.g., ConceptNet [Speer et al., 2017], Quasimodo [Romero et al., 2019], ASCENT [Nguyen et al., 2021]), about products (e.g., Amazon product graph [Dong et al., 2020b], Alibaba e-commerce graph [Luo et al., 2020a]), about commercial enterprises (e.g., Bloomberg KG [Meij, 2019]), and other long-tail domains such as food (e.g., FoodKG [Haussmann et al., 2019]), biomedicine (e.g., Knowlife [Ernst et al., 2014]), culture (e.g., Arco [Carriero et al., 2019]), activity-state-event (e.g., ASER [Zhang et al., 2020]), and more. With the increasing number of knowledge graphs, many efforts have been put into interlinking and aligning them (e.g., [Raad et al., 2020]), to create an unified view of the whole data on the web, which is also referred to as the Linked Open Data[1].

2.1.2 Applications

Knowledge graphs have been used for large commercial search engines and entity-centric question answering systems. For example, Google Search leverages their internal Google KG, and Microsoft Bing uses the Satori KG in their back-end. These internal KGs enable the search engines to give direct answers for look-up queries such as "what is the capital of Germany?" or "who is the president of the USA", for which the answers are "Berlin" and "Joe Biden", respectively. Handling more complex questions is also possible through KG traversal, e.g., "when did Cristiano Ronaldo start playing for Juventus?", or "director of the movie that won the 2021 Oscar?". Various methods have been developed for QA over KGs, by mapping query entities to KG for semantic understanding, KG traversing and ranking to retrieve accurate and concise answers (see [Roy and Anand, 2021] for an overview). KGs are also used to improve the result ranking in document retrieval, by enriching the query and document representations with information about named entities, adding them to the document ranking model [Ensan and Bagheri, 2017, Raviv et al., 2016].

Domain-specific KGs have been used in many advanced commercial products. For example, personal KGs are used in recommendation systems for e-commerce (e.g., Alibaba, Amazon), and in virtual assistants (e.g., Microsoft Cortana, Apple Siri, Amazon Alexa, or Google Home). Commonsense KBs are used for improving visual understanding, i.e., tasks such as object detection in images and videos [Chowdhury et al., 2016], which are beneficial in AI machines and devices like robots, or self-driving cars. Not only that, KGs are also utilized in more complex domains, including academic literature [Wan et al., 2019], finance [Meij, 2019, Albrecht et al., 2019, Reuters, 2017], manufacturing [Bader et al., 2020, Kalayci et al., 2020, Mehdi et al., 2019], and even sensitive domains like law [Junior et al., 2020, González-Conejero et al., 2018] and healthcare [Terolli et al., 2020, Li et al., 2020a, Ernst et al., 2015].

2.1.3 Quantities in KGs

Quantities play an important part of KGs. In many domains (e.g., business or science), interesting facts involve not only entities, but also quantities with numeric values and

[1]https://lod-cloud.net/

units. Quantities capture financial, physical, technological and other important properties of entities such as height of buildings, annual revenue of companies, personal wealth of celebrities, energy consumption of cars, or viscosity of materials. Modern KGs are very rich in entities, containing billions of facts about them. However, when it comes to quantity facts, modern KGs are still very sparse, and this concerns both manually crafted as well as automatically constructed KGs. For example, the height is captured for only less than 0.5% of the buildings in Wikidata, and only 22 out of 9,879 fluids have values for viscosity. Among 3,426 marathon runners available in DBPedia, only 1,346 have their best running time. The incompleteness of quantity knowledge in KGs naturally limits their applicability especially in business-related or scientific domains, leading to wrong or incomplete answers to queries such as "German company with the highest revenue?" or "materials with viscosity lower than that of water?".

Quantities can be represented as values and units. In fact, some KGs already contain a clean and clear quantity schema with rich reference for measures and conversions on which quantity facts could potentially rely. For instance, Wikidata stores not only all possible units for spatial dimensions such as length or volume, but also more sophisticated units from thermodynamics, chemistry or material science. However, in many other KGs, such clean and rich reference is still missing; hence, the quantities are represented just as strings.

2.2 Entity Linking

Entity Linking is the task of linking each named entity mention in a text to a unique entity descriptor to which the mention refers, such as entities from a knowledge graph. Entity Linking is also commonly known as the Named Entity Recognition and Disambiguation task (NERD), which comprises two sub-tasks: Named Entity Recognition (NER) and Named Entity Disambiguation (NED). NER concerns detecting the mention spans in the given input text, while NED targets linking each mention to a unique KG entity. For example, consider the following text:

- *"Jack founded Alibaba in Hangzhou with investments from SoftBank and Goldman."*

The NER task recognizes the four mentions *"Jack"*, *"Alibaba"*, *"SoftBank"* and *"Goldman"*, giving each of them a coarse-grained type label such as `PERSON`, `ORGANIZATION`, `LOCATION`, etc. The NED task disambiguates the mentions and links them to specific KG entities: *"Jack"* $\rightarrow$ *Jack-Ma*, *"Alibaba"* $\rightarrow$ *Alibaba-Group*, *"SoftBank"* $\rightarrow$ *SoftBank-Group* and *"Goldman"* $\rightarrow$ *Goldman-Sachs.*

Entity Linking is an important task in natural language understanding, since it allows

NLP applications to retrieve precise information about the relevant mentions in the input text. By linking entity mentions in text to knowledge graph entities, applications are able to access the rich amount of KG knowledge, including information about entity types, facts between entities, or KGs metadata such as domain, range, constraints, etc.

Named Entity Recognition (NER). Early NER systems are mostly rule-based. They rely on hand-crafted rules in the form of syntactic-lexical patterns for recognizing text spans that denote named entity mentions (e.g., [Mikheev et al., 1999, Krupka and IsoQuest, 2005, Kim and Woodland, 2000]). Some works are based on dictionaries, and operate on specific domains (e.g., [Etzioni et al., 2005, Sekine and Nobata, 2004, Hanisch et al., 2005, Quimbaya et al., 2016]). These rule-based systems perform very well when the lexicon is exhaustive, especially on biomedical, scientific and other specific domains. However, the downside is that rule-based methods usually give low-recall on the results, and cannot be transferred to other domains.

Machine learning for NER has been proposed in both unsupervised and supervised settings. Unsupervised learning systems are typically based on clustering, extracting named entities from clustered groups based on their context similarity (e.g., [Shaalan, 2014]), possibly with the support of a small set of seed rules and patterns (e.g., [Etzioni et al., 2005, Collins and Singer, 1999]). In supervised learning settings, the first NER systems were mostly feature-based, employing feature types such as word-level features (i.e., numeric, case, morphology, POS tags, etc.), lookup features from a gazetteer (e.g., Wikipedia, KGs), document features, corpus features (i.e., multiple occurrences), and more (e.g., [Zhou and Su, 2002, Settles, 2004, Kazama and Torisawa, 2007, Hoffart et al., 2011, Zhu et al., 2005, Ji et al., 2016]). Many machine learning techniques have been used, e.g., Hidden Markov Models (HMM) (e.g., [Bikel et al., 1999]), Decision Trees (e.g., [Szarvas et al., 2006]), Support Vector Machines (SVM) (e.g., [McNamee and Mayfield, 2002]) Maximum Entropy Models (e.g., [Bender et al., 2003, Curran and Clark, 2003]), and especially Conditional Random Fields (CRFs), which have been applied for NER in various domains of input such as biomedical text [Liu et al., 2020b], chemical text [Rocktäschel et al., 2012], social tweets [Liu et al., 2011, Ritter et al., 2011], among others. The main objective is to label each word of the input text with one of the type labels: B- (for beginning) and I- (for inside), or O- for the words that do not belong to any of the named entities.

Recently, state-of-the-art results for NER are usually achieved with deep learning approaches. While BiLSTM-CRF is the most common architecture for this task, the best results are achieved by the Transformer neural architecture, and with a pre-trained language model such as BERT (see [Li et al., 2022] for an overview). Significant performance improvements have been observed in various works (e.g., [Peters et al., 2018, Akbik et al., 2018, Xia et al., 2019, Baevski et al., 2019, Luo et al., 2020b, Liu et al., 2019a, Li et al., 2020c, Li et al., 2020b]).

Named Entity Disambigation (NED). The NED task takes as input a text with recognized mentions from the NER steps, along with the lists of their candidate entites, e.g., *"Jack"* $\rightarrow$ $\{$*Jack-Ma*, *Jack-Nicholson*, *Jack-Black*, ...$\}$, and links each mention to an entity from its candidate list, or to a special candidate "NIL", denoting that the mention is unlinkable. The lists of mention-candidate entities are usually taken from a KB or gazetteer, e.g., constructed from HTML hyperlinks to Wikipedia all over the web.

NED is typically done by collective disambiguation, often using embedding-based features. Approaches for collective disambiguation are based on a common observation that named entity mentions in the same document or related documents are semantically related. Exploiting this, many works for entity disambiguation are graph-based (e.g., [Han et al., 2011, Guo and Barbosa, 2014, Ganea et al., 2016, Rama-Maneiro et al., 2020, Hoffart et al., 2011]). The graphs are typically constructed with vertices are entity mentions, entity candidates and the input document, and edges are the links between them. The edges are weighted, capturing the similarity and coherence between the entity mentions, their candidates and the input text, which are computed from various features such as entity description text, hyperlinks, KG relations and concepts, and more. The objective is to link each mention to a candidate entity in order to maximize the total plausibility of the graph. Various algorithms and techniques for graph inferencing have been used, e.g., [Han et al., 2011, Ganea et al., 2016] use evidence propagation to find the best combination of candidate entities; [Rama-Maneiro et al., 2020] calculates topic coherence and node centrality to find the most relevant entity candidate, avoiding the combinatorial explosion when building candidates graph; [Fang et al., 2019, Yang et al., 2019] proposed a sequential collective inferencing method to disambigate less ambiguous mentions first; [Phan et al., 2019] disambiguates the entity mentions in pairs; [Wei et al., 2019] applies PageRank for candidate selection before the disambiguation; etc.

Similar to the NER task, recent works on the NED task also make use of different embedding-based features to improve performance, e.g., word embeddings, entity embeddings, document embeddings, graph embeddings, and combinations of them (e.g., [Fang et al., 2016, Yamada et al., 2016, Moreno et al., 2017, Le and Titov, 2018, Mueller and Durrett, 2018, Chen et al., 2020, Le and Titov, 2019, Fang et al., 2019, Sevgili et al., 2019]). The main idea is to encode both mentions and their candidate entities into the same latent space using neural networks, to compute the mention-candidate disambiguation confidence scores. The candidates with the highest scores are assigned to the entity mentions. Various neural networks architectures have been used, e.g., LSTM (e.g., [Kolitsas et al., 2018]), CRF (e.g., [Le and Titov, 2018]), BERT (e.g., [Chen et al., 2020]), etc. Moreover, some works use embeddings with advanced indexing mechanism for high-speed and real-time disambiguation (e.g, [Wu et al., 2020a, Parravicini et al., 2019]).

Solutions for NED are not just limited to normal text, but also diverse in their inputs and data features. For example, [Hua et al., 2015] proposed a method optimized for named entity mentions in user queries and blog tweets. As the queries and tweets are very short and hence convey very little information about the entity mentions, the method exploits other features such as entity popularity, entity recency and user interests determined from social interactions. Also working on user tweets, but instead of disambiguating named entity mentions, the method by [Tran et al., 2015] disambiguates hashtags, by leveraging temporal information, entity Wikipedia pages, their edit history and page view statistics.

Entity Linking on Web Tables. Entity linking on semi-structured content like tables from HTML web pages has also been studied intensively in prior work [Limaye et al., 2010, Bhagavatula et al., 2015, Ibrahim et al., 2016, Efthymiou et al., 2017, Ritze and Bizer, 2017, Lehmberg and Bizer, 2017]). Similar to text-based methods, entity linking on web tables usually adopts graph-based techniques with feature engineering that rely on co-occurrences of entity mentions of the table in an external data source (e.g., KB, text corpus), to capture the coherence among entity candidates. These coherence relationships include row-based coherence, column-based coherence, and the coherence between the candidate entities and the table context (i.e., caption, page title, surrounding text, etc.). For example, the pioneer work by [Limaye et al., 2010] introduces five features: tf-idf scores between cell text and entity label, between column header and type label, compatibility between column type and cell entity, between relation and pair of column types, and between relation and entity pairs. [Bhagavatula et al., 2015] computes various features to promote groups of candidate entities that co-occur in Wikipedia pages. Column types and class labels are also used as additional evidence to guide the entity linking process [Mulwad et al., 2010, Zhang, 2017].

Entity mentions are collectively disambiguated into KG entities to maximize the total plausibility of the graph, using various inference algorithms, such as message passing [Mulwad et al., 2013], iterative classification algorithm (ICA) [Bhagavatula et al., 2015], random walks with restart [Ibrahim et al., 2016], and more. [Mulwad et al., 2013] represents the table as a Markov factor graph and uses the semantic message passing algorithm for inferencing. In each iteration, each factor node receives the current assignments from their connected candidate variables and compute agreement between their values. In case outliers are detected, factor nodes send the semantic preferences for the new values back to the candidate variables. Finally, candidate variables update their current assignments, considering the semantic preferences provided from the factor nodes. The whole process is repeated until convergence. The work by [Bhagavatula et al., 2015] employs the iterative classification algorithm (ICA), implementing a very similar idea, where candidate variables update their current values based on the assigned values of other candidates from the previous iteration. The recent work [Ho et al., 2021a] also applies ICA for inferencing, with an extension on incorporating coherence signals from quantitative cues.

Other prominent ideas are also proposed for entity linking on web tables, e.g., [Wu et al., 2016] considers multiple knowledge bases for entity linking, by leveraging "sameAs" relation to reduce errors and ensure good coverage; [Lehmberg and Bizer, 2017] merge tables from the same web page as a single large table to improve entity linking performance; etc.

2.3 Coreference Resolution

Coreference Resolution (CR) is the task of detecting and resolving all mentions in a text or a document that refer to the same individual entities. This plays an important role in downstream applications such as entity linking, chat bots, question answering, relation extraction and more. Essentially, CR can be considered as a complementary task for NERD, as the task recognizes and resolves not only named entities but also their references in the form of nominal or pronominal mentions. For instance, consider the following text:

- *"Ronaldo transferred from Sporting CP to Manchester United in 2003 at the age of 18. At the new club, he won the FA Cup right in his first season."*

While NERD detects and disambiguates the four named entities *"Ronaldo"*, *"Sporting CP"*, *"Manchester United"* and *"FA Cup"*, CR additionally recognizes *"he"* and *"his"* as the coreferences of *"Ronaldo"*, and *"the new club"* as the coreference of *"Manchester United"*. In contrast to NERD, coreference resolution does not disambiguate the mentions and link them to the KG; instead, the task only works on the surface forms of the mentions, grouping them into clusters of the same entities.

Methods for coreference resolution typically start with mention detection, similar to the Named Entity Recognition (NER) task. In addition to NER, mentions for the CR task also include nominal (e.g., "the new club") and pronominal phrases (e.g., "he", "his"). Various techniques for mention detection have been proposed, e.g., rule-based, statistical-based, and deep-learning based approaches (see [Lata et al., 2022] for an overview).

The core of CR is to group mentions into clusters of the same entity targets. Approaches can be classified into several categories: mention-pair models, mention-ranking models, entity-based models, latent-structure or language-modeling models. Methods from these approaches are often combined.

Mention-pair models are the earliest and simplest solutions for coreference resolution, which typically operate on a pair of mentions at a time to generate a binary score, denoting the likelihood that the two mentions are coreferences of each other (e.g., [Soon et al., 2001, Ng and Cardie, 2002, Denis and Baldridge, 2007]). These methods often leverage feature engineering to compute the coreference scores: features such as POS tags, dependency tree or advanced feature selection techniques like named entity clustering, dictionary/alias searching, or other similarity measures [Charton and Gagnon, 2011].

The most obvious weakness of mention-pair models is that they do not consider the dependency between a mention and other candidate antecedents during the scoring. Mention-ranking models overcome this disadvantage, by simultaneously ranking the candidate antecedents during the scoring of mention pairs (e.g., [Yang et al., 2003, Chang et al., 2013, Wiseman et al., 2015, Lee et al., 2017]).

Entity-based models continue to improve over mention-pair and mention-ranking models by considering the transitivity between candidate mentions, which enables the resolution with long-dependency in the input (e.g., [Luo et al., 2004, Yang et al., 2004, Ratinov and Roth, 2012]). Many of them are extensions and adaptations of mention-ranking methods, by integrating global features using neural learning techniques, to allow the ranking and merging of mention clusters (e.g., [Wiseman et al., 2016, Clark and Manning, 2016]). External entity-centric resources such as Wikipedia, DBPedia and YAGO have also been integrated for improving the quality of coreference resolution (e.g., [Plu et al., 2018, Prokofyev et al., 2015]). Several works target to optimize the running time and memory footprint of the resolution (e.g., [Toshniwal et al., 2020, Xia et al., 2020]).

Similar to many other NLP tasks, most state-of-the-art results for coreference resolution have been built on deep learning techniques. These methods are based on the mention-ranking and entity-based approaches, with improvement from using neural learning techniques. A prominent example is the AllenNLP coref system [Lee et al., 2018], which has employed a gated attention mechanism and a span-ranking technique together with the pretrained language model ELMo to construct a latent antecedent trees, which can be further converted into clusters of coreferences. Several other works have been proposed based on the similar idea, using various techniques such as adversarial training (e.g., [Subramanian and Roth, 2019]), reinforcement learning (e.g.,[Aralikatte et al., 2019]), entity equalization (e.g.,[Kantor and Globerson, 2019]), graph neural networks (e.g.,[Liu et al., 2020a]), etc. Finally, the recent work [Wu et al., 2020b] has proposed the novel CorefQA method for solving CR task, by converting it into question answering style, leveraging the SpanBERT language model.

2.4 Quantity Recognition

Prior work has looked into recognizing and extracting numeric expressions such as *"50 km/h"* or *"€10 million"* from web content, interpreting them as quantities with a value and a unit (e.g., [Roy et al., 2015, Alonso and Sellam, 2018, Sarawagi and Chakrabarti, 2014, Banerjee et al., 2009]). Most of them are based on rules. For example, [Banerjee et al., 2009] proposed a rule-based system for detecting quantities for the purpose of ranking search results involving quantities. The quantity detector consists of 150 rules run on the rule-based JAPE engine of the GATE NLP package[2], covering quantities from various domains such as length, mass, power, mileage, speed, density, volume, area, money, duration, time epoch, temperature and more. However, their rules are designed specifically for the used benchmark queries, and do not generalized to the whole English grammar.

[Roy et al., 2015] proposed a learning method for extracting quantities from text, consisting of two steps: segmentation and standardization. The segmentation steps aim at recognizing the continuous segments of quantities in text, using CRFs and classifiers [Punyakanok and Roth, 2000]. The models are trained with various features, including word class features (e.g., dictionaries of units, written numbers, month names, temporal words), character-based features (e.g., digits, suffixes), and POS tags. The second step standardization normalizes the segmented text into quantity value and unit, using rules, e.g., changing numbers in string form into floating point values, dates into internal date type, rewriting known units to a standard unit, and so on. The system also captures bounding and approximation indicators such as "more than", "less than", "nearly", "about", etc.

[2]https://gate.ac.uk/

Recognizing quantities in web tables has been addressed in the work QEWT [Sarawagi and Chakrabarti, 2014]. In contrast to quantities in text, quantities in web tables may have constituents appearing in different places: values appear in the table body, units and scaling factors appear in the headers, or sometimes in the table caption. [Sarawagi and Chakrabarti, 2014] proposed a probabilistic context free grammar (CFG) for extracting units from table headers. For example, given the header *"Net profit/(loss) (£m)"*, the extractor recognizes "British pound" as the unit and "million" as the scaling factor. The extractor is feature-based, by harnessing co-occurrence statistics, frequency indicators, dictionary matching as well as other rules. QEWT also proposed an unit catalog called QuTree, consisting of 44 quantity types (e.g., length, area, speed), 750 units and their full names (e.g., kilometre per hour), symbols (e.g., kmph, km/h) and lemmas. By mapping the recognized units to QuTree, the QEWT system is able to perform simple reasoning over quantities such as converting, comparing or analyzing the quantity distribution. An alternative unit catalog is also constructed by [Ibrahim et al., 2016].

Other tools for recognizing and manipulating quantities include Quantulum3[3], Pint[4], Natu[5], and so on.

2.5 Open Information Extraction

Open Information Extraction is an IE paradigm that turns unstructured text into relational tuples, consisting of a set of arguments and a phrase that describes the semantic relation between them. In the simplest form, Open IE tuples are in the form of triples ⟨*arg0, relation, arg1*⟩. For example, from the sentence "Joe Biden is the president of the US.", Open IE extracts the tuple ⟨*Joe Biden; is the president of; the US*⟩. Advanced OpenIE systems also extract higher-arity tuples ⟨*arg0, relation, arg1, arg2, ...*⟩, in which *arg0* and *arg1* act as the subject and object, while other arguments are facets/qualifiers, adding more information to the extraction. For example, from the sentence "The BMW i8 has price of 138k Euros in Germany", the following tuple is extracted: ⟨*The BMW i8; has; price of 138k Euros; in Germany*⟩.

OpenIE is different from traditional information extraction paradigms in various aspects. Traditional IE paradigms often rely on pre-defined relations with a fixed set of extraction rules and patterns from a narrow and specific domain. These patterns are typically either hand-crafted, or being learned from a set of human-annotated training examples. Although achieving high-quality extractions, traditional IE methods do not

[3]https://pypi.org/project/quantulum3/
[4]https://pint.readthedocs.io/
[5]http://kdavies4.github.io/natu/

generalize to new domains, in which the user has to define not only the new input relations but also their extraction rules and/or new training examples. This process is time-consuming and requires extensive human involvement. In contrast, OpenIE does not have these limitations. The extraction paradigm targets to extract all kinds of relations in text, by relying on unsupervised extraction strategies, treating them as normal phrases. OpenIE is automatic, fast, domain-independent, and scales to large heterogeneous text corpora. Hence, this extraction paradigm has the potential do handle the vast amount of data on the web. However, OpenIE systems also have drawbacks: their outputs are still mere strings, and hence, the extracted statements cannot be directly added to KGs as facts, as their SPO arguments are left non-canonicalized.

A large number of OpenIE systems have been proposed (e.g., TextRunner [Banko et al., 2007], ReVerb [Fader et al., 2011], OLLIE [Mausam et al., 2012], ReNoun [Yahya et al., 2014], ClausIE [Corro and Gemulla, 2013], SrlIE [Christensen et al., 2010], BONIE [Saha et al., 2017], CalmIE [Saha and Mausam, 2018]). Many of them employ shallow linguistic features such as part-of-speech (POS) tags or dependency tree for fast extraction.

OpenIE systems can be classified into several groups: learning-based systems, rule-based systems, clause-based systems, and systems that capture inter-proposition relationships.

Learning-based systems. TextRunner [Banko et al., 2007] was the first OpenIE system falling into this category. The learning-based system consists of three modules. First, a self-supervised learner is built to model the relation trustworthiness of tuples, by relying on shallow linguistic features including POS tags and noun phrase (NP) chunks. The second module, single-pass extractor, is run over the input text to identify pairs of NP as the tuple arguments, and heuristically examine the words between the NP pairs to select the relevant ones into the relation phrases. The learned trustworthiness model is then used to classify and keep only trustworthy candidate tuples. Finally, a redundancy-based assessor runs through the resulting tuples to merge the identical ones, assigning each a probability confidence score that reflects how frequently the extraction was found.

[Wu and Weld, 2010] proposed the two systems WOE^{pos} and WOE^{parse}, with similar idea. To examine the relation between two noun phrases, WOE^{pos} employs a linear-chain Conditional Random Field (CRF) to recognize the text denoting the relation. The CRFs is trained with distant supervision, by matching attributes from Wikipedia infoboxes against their corresponding article sentences. WOE^{parse}, instead, relies on the shortest paths between NPs on the dependency tree to express the relations, and hence can extract tuples with long-range dependencies in the input text.

The OLLIE system [Mausam et al., 2012] follows the same idea of bootstrapping for learning pattern templates from dependency tree, and extends the previous approach by adding a context-analysis step, to expand the extracted tuples with attribution and clausal modifiers. Moreover, OLLIE learns not only verb-based relations, but also relation patterns that are mediated by nouns and adjectives.

Rule-based systems. The ReVerb system [Fader et al., 2011] makes use of hand-crafted extraction rules to extract OpenIE tuples. These rules are in the form of syntactic constraints, in particular, POS tag-based regular expressions. This way addresses the three common extraction errors in the previous systems, namely uninformative extractions (i.e., important information are discarded), incoherent extractions (i.e., unmeaningful extractions), and overly-specific relations (i.e., too much redundant information). Moreover, in contrast to previous OpenIE systems, ReVerb is a relation-centric extractor. Instead of starting by determining argument pairs, ReVerb first identifies the verb-phrases as relations, then find the appropriate NP arguments for each relational phrase.

Syntactic rules have also been enforced on dependency parse tree for OpenIE in many other works (e.g., KrakeN [Akbik and Löser, 2012], Exemplar [de Sá Mesquita et al., 2013], PropS [Stanovsky et al., 2016]). KrakeN is able to capture complete facts from sentences, extracting tuples of arbitrarily length. Exemplar is based on Semantic Role Labeling task to assign each of the tuple arguments a specific role. PropS does not work directly on the dependency tree, but rather transforms it into a proposition structure, produced by a rule-based converter.

Clause-based systems. The main idea of the ClausIE system [Corro and Gemulla, 2013] for extracting OpenIE tuples is to map the dependency relations of the input text to clause constituents, exploiting the fact that English clauses can be classified into only seven different types, according to the grammatical function of their constituents. Based on the type, one or more tuples can be generated from each clause, expressing different pieces of information.

The Stanford OpenIE system [Angeli et al., 2015] also extract clause-based tuples. In contrast to ClausIE, Stanford OpenIE recursively traverses down of the dependency tree and predicts whether the sub-tree at each node forms an independent clause or not. Resulting clauses are shortened to maximize their usefulness before the actual tuples are extracted from them by using a small set of 14 hand-crafted rules.

Systems capturing inter-proposition relationships. Recent state-of-the-art OpenIE systems target extracting tuples with context, which have been largely overlooked by previous systems. Context might be a condition in which the proposition holds, e.g.,

(⟨*Romney; will be elected; President*⟩; `ClausalModifier` *if; he wins five key states*)

or a hypothesis, e.g.,

(⟨*the earth; be the center of; the universe*⟩; `AttributedTo` *believe; Early astronomers*)

or even a tuple, e.g.,

#1: ⟨*The Embassy; said; that; #2*⟩

#2: ⟨*6,700 Americans; were; in Pakistan*⟩

Systems such as OLLIE [Mausam et al., 2012], SrlIE [Christensen et al., 2010], ReNoun [Yahya et al., 2014], CalmIE [Saha and Mausam, 2018] and NestIE [Bhutani et al., 2016] fall into this category.

Open information extraction for quantities. The most prominent OpenIE system that handles quantities is BONIE [Saha et al., 2017], which extracts tuples where one of the arguments is a number or a quantity-unit phrase. Similarly to other works, BONIE employs a bootstrapping technique to learn dependency patterns for extracting numerical relations. The novelty of BONIE is its ability to extract implicit relation phrases. For example, given the input text "Donald Trump is 70 years old", BONIE extracts the tuple ⟨*Donald Trump; has age of; 70 years*⟩, while previous OpenIE systems only yield ⟨*Donald Trump; is; 70 years old*⟩. Hence, BONIE outputs are more meaningful for quantity-related downstream tasks.

Chapter 3

Qsearch: Answering Quantity Queries from Text

3.1 Introduction

3.1.1 Motivation

Quantities are common in search queries, for example to find a product within a specific price range, cars or mobile phones with desired technical or environmental properties, or athletes who ran a race in a certain time. When a user issues a quantity search query, such as *"Hybrid cars with price under 35,000 Euros"*, she expects the search engine to understand the quantities and to return relevant answers as a list of entities. However, Internet search engines treat quantities largely as strings ignoring their values and unit of measurements. As a result, they cannot handle numeric comparisons and ultimately fail. QA systems based on knowledge graphs also fail, as their coverage of quantitative facts is very limited.

This chapter sets out to provide support for answering quantity queries from one major type of web content, namely text, over a wide variety of expressive measures, to overcome this severe limitation of today's search engines and knowledge graphs. Our method extracts quantity-centric structure from Web contents, uncovering the hidden semantics of linking quantities with entities.

3.1.2 Problem Statement

We define our problem as follows. Given a quantity query and a corpus of text pages, find a ranked list of entities that match the given query. A quantity query is a triple (t^*, q^*, X^*), where t^* is the semantic type of the expected answers, q^* is a quantity-centric search condition, and X^* is the context that connects the entity type t^* with quantity condition q^*. For example, for the query *"Cars with price less than €35,000 in Germany"*, the triple (t^*, q^*, X^*) is: (*cars;* $<$ *€35.000;*{*price, Germany*}). Our problem has two dimensions. The first is to understand the content of the text snippets and extract the relevant quantity facts. The second is to match such extracted assertions (inevitably with noise and errors) against a query and compute a ranked list of relevant entity answers.

3.1.3 Approach

This chapter presents Qsearch, an end-to-end system for answering quantity queries. Qsearch employs a deep neural network to extract quantity facts from text, this way lifting textual information into semantic structures. Then, it utilizes a statistical matching model to retrieve and rank answers.

We model the first component, quantity fact extraction, as a Semantic Role Labeling (SRL) task [Gildea and Jurafsky, 2002] and devise a deep learning method to label words in the sentences with relevant roles. We label each word as entity, quantity or context (or other). Then we use these tags to extract quantity fact triples in form of *(entity, quantity, context)*. For the second component, query matching, we devise a novel matching method to retrieve a ranked list of relevant entities that answer the user's quantity query.

3.1.4 Contribution

The salient contributions of this work are as follows:

- We present Qsearch, a system for answering quantity queries from text.
- We propose a deep neural network for quantity fact extraction, and a matching model for answering quantity queries.
- We present extensive experiments on benchmark queries collected by crowdsourcing.

We make the experimental data and code available to the research community[1].

[1]https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/quantity-search/qsearch

3.2 Computational Model and System Overview

In this section, we introduce the computational model for our approach and give an overview of the Qsearch system and its components.

3.2.1 Model for Facts, Queries and Answers

Extraction Model. The *input* of this model is a corpus of text documents $\mathcal{T}$ with text snippets (e.g., sentences or paragraphs) that contain entity and quantity mentions.

The *output* of this model is a set of *quantity facts* extracted from the text corpus, $\mathbb{F} = \{\mathcal{F}_1, \mathcal{F}_2, ...\}$, where a quantity fact is defined as follows.

Definition 3.2.1 (Quantity fact). *A quantity fact (Qfact) is a triple $\mathcal{F} = (e, q, X)$, where:*
- e is an entity;
- $q = (v, u, r)$ is a quantity consisting of a numerical value v, a canonicalized unit u (e.g., km, \$) and a value resolution r (exact, approximate, upper/lower bound, interval);
- $X = \{x_1, x_2, ...\}$ is a context, which is a bag of words describing the relation between e and q.

Example 3.2.1. Given the text snippet *"BMW i8 costs about 138k Euros in Germany and has a battery range between 50 and 60 km."*, we can extract the following Qfacts:
- $\mathcal{F}_1 : e =$ *BMW i8*; $q =$ *(138.000, €, approximate)*; $X = \{$*costs, Germany*$\}$
- $\mathcal{F}_2 : e =$ *BMW i8*; $q =$ *(50-60, km, interval)*; $X = \{$*range, battery*$\}$ □

The Qfact representation is similar to the RDF model [Lassila and Swick, 1999], which represents each fact as a *(subject, predicate, object)* triple. In the Qfact model, the entity e and the quantity q correspond to the *subject* and the *object*, respectively. The context X in Qfacts is a proxy for the *predicate* in the RDF model. However, it differs in two essential points: first, the context X can capture more than one relation between e and q; second, the context X consists of a set of non-canonicalized tokens, instead of a unique canonicalized predicate in a knowledge graph.

This relaxed representation is a judicious design choice and essential for the flexibility of our approach: first, we can represent complex n-ary facts using a simple Qfact triple; second, our model can generalize to unseen relations; third, our model can cope with the inevitable diversity and uncertainty in the language expressions of the underlying text snippets. In theory, it is conceivable that all arguments that appear in the context X are also individually extracted and canonicalized to fill the slots of a frame-like structured record. However, approaches along these lines do not work robustly and suffer from heavy propagation of noise and errors.

The Qfact model allows different representations of the same fact, and the underlying text corpus may express the same knowledge by different paraphrases. Hence, Qfacts are more expressive towards answering queries via approximate matches and related phrases.

Matching Model. The *input* of this model is a set of Qfacts $\mathbb{F} = \{\mathcal{F}_1, \mathcal{F}_2, ...\}$ extracted from the text corpus, and a *quantity query* $\mathcal{Y}$ defined as:

Definition 3.2.2 (Quantity query). *A quantity query (Qquery) is a triple* $\mathcal{Y} = (t^*, q^*, X^*)$ *where:*
- t^* *is the semantic type of the target answers;*
- $q^* = (v, u, o)$ *is a quantity condition consisting of a numerical value* v*, a canonicalized unit* u *(e.g., km, \$) , and a comparison operator* o *(exact, approximate, upper/lower bound, interval);*
- $X^* = \{x_1, x_2, ...\}$ *is a context condition, expressed by a bag of words that describes the relation between* t^* *and* q^**.*

Example 3.2.2. Given the query *"Cars with price less than 100k Euros in Germany"*, its corresponding Qquery is as follows:
- $\mathcal{Y}$: t^* = *car*; q^* = *(100.000, €, upper bound)*; X^* = {*price, Germany*} □

Each part of a Qquery imposes a constraint on its counterpart in a Qfact considered as a candidate answer.

Definition 3.2.3 (Query answer). *A Qfact* $\mathcal{F} = (e, q, X)$ *is an answer for a Qquery* $\mathcal{Y} = (t^*, q^*, X^*)$ *iff (1)* e *is an entity of type* t^**, (2) the quantity* q *satisfies the quantity condition* q^* *and (3) the context* X *(approximately) matches the context condition* X^**.*

Example 3.2.3. Consider the Qquery in Example 3.2.2 and the two text segments *"German dealers sell the BMW X3 at a price as low as 55,000 Euros"* and *"Car dealers in Munich sell the BMW X3 starting at 55,000 Euros"*. The Qfact extracted from the first snippet with e = *BMW X3*, q = *(55.000, €, lower bound)*, and X = {*German, dealers, sell, price*} is a strong match for the query; whereas the Qfact extracted from the second snippet with e = *BMW X3*, q = *(55.000, €, lower bound)*, and X = {*car, dealers, Munich, sell*} is an approximate match (by embedding-based relatedness). □

The *output* of this model is a ranked list of entities $\mathcal{E}^* = \{e_1, e_2, e_3, ...\}$ from matching Qfacts with the Qquery, which will be discussed in Section 3.4.

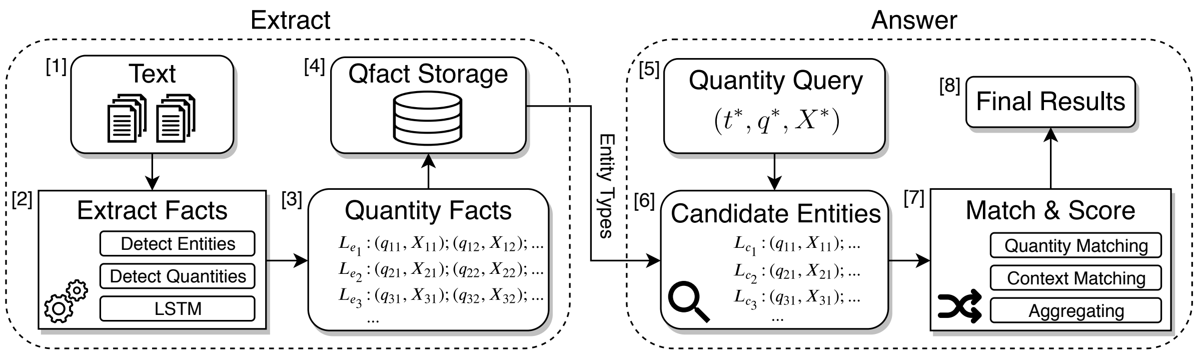


Figure 3.1: Overview of Qsearch.

3.2.2 The Qsearch System

Figure 3.1 gives an overview of the architecture of Qsearch. The arrows in the figure depict information flow between the different system components. Qsearch consists of two main stages: *Extract* and *Answer*.

Extract. We preprocess the text corpus (Block 1) to recognize and disambiguate named entities and link them to an external knowledge base (KB). We also identify mentions of quantities in the text and normalize them into standard units. Subsequently, we run a deep neural network to extract Qfacts from the preprocessed text (Block 2). We learn and employ a specifically designed Long Short Term Memory (LSTM) network, which will be described in Section 3.3.

Extracted Qfacts are organized and grouped by their named entities, such that each individual entity e_i is mapped to a list of quantities and related contexts $L_{e_i} = \{(q_{i1}, X_{i1}), (q_{i2}, X_{i2}), ...\}$ (Block 3). All extracted Qfacts are stored in a data repository (Block 4, based on Elasticsearch in our implementation), where entities are linked to their semantic types from the KB.

Answer. We answer incoming Qqueries by matching them against the Qfacts from the *Extract* stage. For a Qquery (t^*, q^*, X^*) (Block 5), we first apply an entity-type filter, eliminating entities with the wrong type. This results in a set of candidate entities $\mathcal{C} = \{c_1, c_2, ...\}$ (Block 6) satisfying the type constraint t^*, along with their quantity-context pairs $\{L_{c_1}, L_{c_2}, ...\}$. In Block 7, we discard all candidate answers that do not satisfy the quantity condition q^*. Finally, we compute a matching score for each candidate entity $c \in \mathcal{C}$ based on the contexts X in the quantity-context pairs L_c, using a statistical language model or a text embedding method, which will be described in Section 3.4. The candidate entities are ranked by their scores and returned to the user (Block 8).

In the following Sections 3.3 and 3.4, we discuss in detail the Qfact extraction model and the matching and answering model, respectively.

3.3 Quantity Fact Extraction From Text

In this section, we describe our method for extracting Qfacts from natural language text. At the core of our solution is a deep-learning neural network for sequence tagging, running on individual sentences.

3.3.1 Input Preprocessing

In the first step, we preprocess the input text corpus by detecting entities and quantities appearing in each individual input sentence. We perform Named Entity Disambiguation (NED) using the AIDA [Hoffart et al., 2011] system, which links named entities to the YAGO knowledge base [Suchanek et al., 2007]. To achieve a better detection quality, we run NED on a per-document instead of per-sentence basis. For detecting quantities, we make use of the Illinois Quantifier [Roy et al., 2015], a state-of-the-art tool for recognizing numeric quantities in text, along with some hand-crafted rules (e.g., regular expressions). Subsequently, each identified quantity is replaced by a placeholder "*_QT_*".

Example 3.3.1. Input and output of this preprocessing step look as follows:

$$\begin{array}{l|l}
\textit{sentence} & \textit{BMW i8 has price of 138k Euros in Germany and range from 50 to 60 km on battery .} \\
\textit{preprocessed} & \overbrace{\textit{BMW i8}}^{e_1=\textit{<KB:BMW\_i8>}} \textit{ has price of } \underbrace{\textit{\_QT\_}}_{q_1=\textit{(138.000,€,appr.)}} \textit{ in } \overbrace{\textit{Germany}}^{e_2=\textit{<KB:Germany>}} \textit{ and range } \underbrace{\textit{\_QT\_}}_{q_2=\textit{(50-60,km,interval)}} \textit{ on battery .}
\end{array}$$

□

3.3.2 Sequence Tagging Model

In the second step, we aim to extract complete Qfacts from the preprocessed sentences. For each quantity detected in the previous step, we want to identify the entity to which it refers and the relevant context tokens that express the entity-quantity relation.

Example 3.3.2. Consider the preprocessed sentence in Example 3.3.1. If we use the first quantity q_1 = *(138.000, €, approximate)* as the input's pivot, we want to obtain the output $e(q_1) = e_1 =$ *<KB:BMW_i8>* and $X(q_1) = \{$*price, Germany*$\}$. Analogously, with q_2 = *(50-60, km, interval)* as pivot, the desired output is $e(q_2) = e_1 =$ *<KB:BMW_i8>* and $X(q_2) = \{$*range, battery*$\}$. □

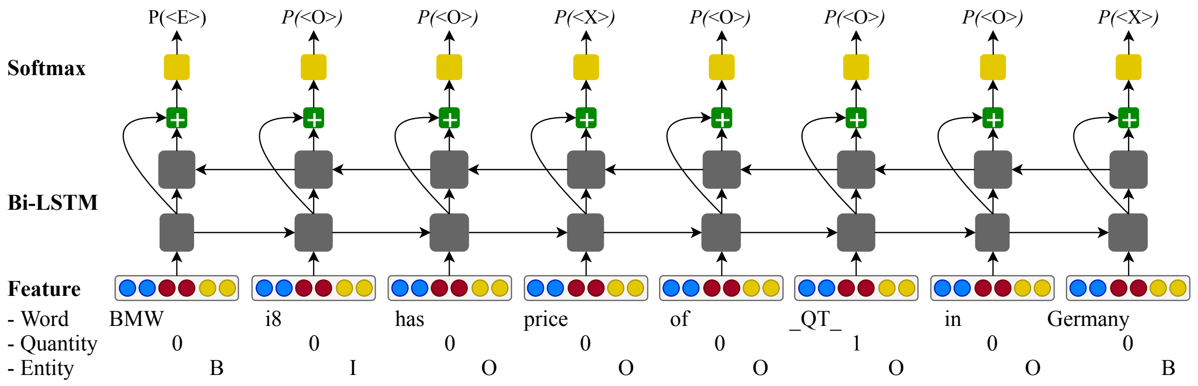


Figure 3.2: The Qfact extraction model used by Qsearch.

We formalize this task as a sequence labeling problem as follows.

Task 1 (Quantity Fact Extraction). *Given a preprocessed sentence S with the set of detected entities $\mathcal{E} = \{e_1, e_2, ...\}$, the set of detected quantities $\mathcal{Q} = \{q_1, q_2, ...\}$ and a selected pivot quantity of interest $q_i \in \mathcal{Q}$, the task of quantity fact extraction is to label each token of the sentence with one of the following tags: (i) $<E>$, for denoting the entity that q_i refers to; (ii) $<X>$, for denoting the context tokens that relate q_i and its entity; and (iii) $<O>$, for all other tokens.*

Our problem resembles the Semantic Role Labeling task [Gildea and Jurafsky, 2002], which is typically addressed by Conditional Random Fields (CRFs) or Long Short Term Memory (LSTM) models. Figure 3.2 depicts the bi-directional LSTM model that we devised for this task, inspired by prior work [He et al., 2017]. Other models for sequence labeling (e.g., [Zhou and Xu, 2015, FitzGerald et al., 2015]) could be easily incorporated as well. Our labeling network consists of three layers: Input Features, Bi-LSTM, and Softmax. While this general architecture is close to any other LSTM model, the most unique point here is the input representation, as described next.

Input Features. Each token of the preprocessed input sequence is represented as the concatenation of three input feature vectors:

1. *Word*: We include word embeddings as an input feature, which enables the neural model to generalize to different words having similar meanings. In our implementation, we use Glove [Pennington et al., 2014] precomputed embeddings.

2. *Quantity*: We provide the position of the pivot quantity to the model as input. When sentences contain multiple quantities (which is a relatively frequent case),

our model operates one quantity at a time and we re-run the model for different quantities.

3. *Entity*: We also provide information about the recognized entities as input to the neural model. As entities often span multiple tokens, we employ the BIO tagging mechanism [Ramshaw and Marcus, 1995], where a tag B is used for tokens at the beginning of an entity name, I for tokens inside the name, and O for other tokens. With this representation, the output of the model only needs to tag the first token of a multi-word entity name with $<E>$, and subsequent tokens are tagged with $<O>$. Figure 3.2 shows an example: *"BMW i8"* is chosen as the entity connected with the pivot quantity; only the first token *"BMW"* is tagged as $<E>$ in the output.

3.3.3 Output Constrained Decoding

The output of the model are the probabilities of each token word in the input belonging to each of the three tags $<E>$, $<X>$ and $<O>$, produced by the Softmax layer. In neural models, usually the tag with the highest score will be assigned to each token word. However, this standard technique would not take into account the dependencies between output tags, and hence might give us an invalid tag sequence. To solve this issue, we impose the following two constraints on the output of the model at decoding time, and find the most probable tag sequence satisfying them: *(i)* only one tag $<E>$ can appear in the output (namely, for the one entity to which the pivot quantity refers); and *(ii)* that tag $<E>$ has to be at the start token of an entity name.

To find the most probable tag sequence, we use Dynamic Programming to decode from left to right. Specifically, we compute subsequences of tags $Seq_{i,j,k}$ for every $i \in \{1..n\}$ (n is the sentence length); $j \in \{<E>, <X>, <O>\}$; and $k \in \{0, 1\}$. Here, $Seq_{i,j,k}$ denotes tag subsequence with the highest probability for tokens from position 1 to position i, where the tag of token at position i is j, and the subsequence contains k $<E>$ tags. Note that the probability of a tag subsequence is computed as the product of the probabilities of its constituent tags. The final tag sequence can be derived at $i = n$.

3.3.4 Distant Supervision Training

As training data is an important factor but difficult to obtain, and manual labeling at scale is too expensive, we employ distant supervision to generate training data for the Qfact extraction model. We use unsupervised, pattern-based Open Information Extraction (Open IE) to overlay an n-tuple structure (with triples or higher-arity tuples) on the input text. We employ the OpenIE4 tool [Mausam, 2016] to this end, and then use its output tuples to generate training data. This process consists of two steps:

- *Step 1: Capture information areas:* We define an *information area* as a subset of tokens from a sentence, which presents complete information about a fact. We run Open IE on the *unprocessed* sentence to detect all possible tuples expressed by the text. Each of these tuples has a confidence score; to ensure the quality of the generated training samples, we only keep tuples having a confidence score of at least 0.9. Each of the selected tuples corresponds to an information area.

Example 3.3.3. Consider the unprocessed sentence in Example 3.3.1, suppose the following tuples are extracted by Open IE: (1): *(BMW i8; has; price of 138k Euros; in Germany)*$^{0.95}$, (2): *(BMW i8; has; range from 50 to 60 km on battery)*$^{0.9}$, (3): *(BMW i8; has; price of 138k Euros)*$^{0.8}$, (4): *(BMW i8; has; range from 50 to 60 km)*$^{0.5}$, and (5): *(BMW i8; has; price)*$^{0.1}$. We only keep high-confidence tuples (1) and (2), which contain complete information. Then the following information areas are chosen for training:

$$\overbrace{\underbrace{\textit{BMW i8 has}}_{(2)}\textit{ price of 138k Euros in Germany}}^{(1)}\textit{ and }\underbrace{\textit{range from 50 to 60 km on battery}}_{(2)}. \quad \square$$

- *Step 2: Transform infomation areas into training samples:* We map the information areas obtained in Step 1 with entities and quantities detected from the preprocessing phase:

$$\overbrace{\underbrace{\textit{<KB:BMW\_i8> has}}_{(2)}\textit{ price of }\_QT\__{(1)}\textit{ in <KB:Germany>}}^{(1)}\textit{ and }\underbrace{\textit{range }\_QT\__{(2)}\textit{ on battery}}_{(2)}.$$

$\square$

With this mapping, information areas yield training samples for the neural network. We apply conservative filters so that this self-training process minimizes spurious samples. First, we keep only information areas that contain exactly one quantity $\_QT\_$, the pivot quantity. Second, since English sentences tend to express quantity information in active voice, the entity connected to the pivot quantity should appear in the first argument (subject) of the Open IE tuple. For instance, information area (1) has two entities *<KB:BMW_i8>* and *<KB:Germany>*; we choose the former as the one to which quantity $\_QT\__{(1)}$ refers. Finally, we discard all information areas where the subject of the Open IE tuple contains more than one entity.

At this point, for each information area, we have a quantity and a unique entity to which it refers. The context between them is determined from the remaining tokens in the information area based on their Part-of-speech (POS) tags. We allow only the following POS patterns to form the context: noun (NN*), verb (VB*), adjective (JJ*), adverb (RB*), and foreign word (FW, to capture out-of-vocabulary names). We also use pre-defined stopwords to remove uninformative tokens from the context. The resulting Qfact, along with the *<E>,<X>,<O>* tags for its token sequence, becomes a positive training sample. As negative training samples, we collect all information areas where no entity could be identified to relate with the pivot quantity, i.e., all tokens are tagged as *<O>*.

3.4 Candidate Fact Matching Model

This section describes our method to answer Qqueries from the extracted Qfacts. To this end, each Qfact is assigned a score denoting its relevance to the given Qquery.

3.4.1 Query Parsing

Input questions are mapped into Qqueries by a rule-based parser for recognizing answer type and quantity condition; all other tokens (except stopwords) are included in the query context. The parser uses a dictionary of YAGO types and a dictionary of quantity units. An alternative to this rule-based technique would be to apply the same neural extraction method to questions that we have used to extract Qfacts from text. However, the questions are easier to handle, and the rule-based parser works well.

Task 2 (Quantity Fact Scoring). *Given Qquery* $\mathcal{Y} = (t^*, q^*, X^*)$ *and Qfact* $\mathcal{F} = (e, q, X)$*, compute a distance score* $d(\mathcal{F}, \mathcal{Y})$ *reflecting the relevance of* $\mathcal{F}$ *regarding* $\mathcal{Y}$*.*

Without loss of generality, we assume that a lower score denotes a better fact. $\mathcal{F}$ should be a high-ranked answer for $\mathcal{Y}$ iff the following three conditions hold: (1) e is an entity of type t^*, (2) q satisfies q^*, and (3) X is a good (approximate) match for X^*.

3.4.2 Entity - Type Matching

We only consider the Qfact $\mathcal{F}$ if the entity e has type t^*. Since the entities from text are linked to an external knowledge base, we make use of the type information from the KB to filter out unsuitable facts for $\mathcal{Y}$.

3.4.3 Quantity Matching

We also discard $\mathcal{F}$ if q does not satisfy q^*. This is the case when either (1) the units of q and q^* relate to different concepts (e.g. *km* (length) vs. €(money)) and are thus incomparable; or (2) their values (after conversion to the same unit) do not match the comparison operator of q^*. Since quantity matching is not the focus of our paper, we apply a simple matching method as follows. First, we use hand-crafted rules for unit conversions, re-scaling if needed (e.g., for kilo, mega, etc.), and value normalization. Second, we turn the quantity value into an interval based on its resolution. For example, when the query is about approximate matches, a quantity value v is smoothed into the interval $[v-\delta, v+\delta]$ with a configuration parameter δ. In experiments, we set δ to 5% of v. A comparison is considered a match when the two intervals overlap, and their units (after conversion and re-scaling) match.

3.4.4 Context Matching

If the Qfact $\mathcal{F}$ satisfies the above two constraints, we will consider the similarity between the query context X^* and the fact context X. We propose to use the following two approaches for measuring the context relevance: a *probabilistic* and an *embedding-based* approach.

Probabilistic Ranking Model. We adopt the Kullback-Leibler (KL) divergence between the query context X^* and the fact context X, which is typically used in statistical language models [Zhai, 2008]. The scoring function is defined as follows:

$$\begin{aligned} d(\mathcal{F}, \mathcal{Y}) = KL(X^*, X) &= H(X^*, X) - H(X^*) \\ &\equiv H(X^*, X) = -\sum_{w \in \mathcal{V}} P(w|X^*) \log P(w|X) \end{aligned}$$

where $\mathcal{V}$ is the word vocabulary, $H(X^*)$ is the entropy of X^*; $H(X^*, X)$ is the cross entropy between X^* and X; and $\equiv$ indicates rank equivalence (i.e., preserving order). Since we are only interested in ranking fact contexts in response to a query context, we can omit $H(X^*)$. The word probability $P(w|X^*)$ for the query context is estimated using Maximum Likelihood Estimation (MLE) on an expanded version X^*_E of X^* as:

$$P(w|X^*) = count(w \in X^*_E)/|X^*_E|$$

To expand a query context, we resort to WordNet [Miller, 1995] and add all synonyms of the context words to it. For the fact context, we estimate the word probability $P(w|X)$ using Jelinek-Mercer smoothing as:

$$P(w|X) = (1 - \lambda) \times count(w \in X)/|X| + \lambda \times P(w|B)$$

This linearly combines the MLE from the fact context X with the MLE obtained from a background corpus B. The smoothing parameter λ (set to $\lambda = 0.1$ in our system) controls the influence of the background corpus on the probability estimate. We construct the background corpus B from all sentences of the entire text corpus that contain at least one quantity (total 39M sentences in our data).

Embedding-based Ranking Model. We observed on our data that the query context X^* is often shorter than the fact context X, since sentences are often more verbose than the typically short queries. Hence, to measure the distance score of X with regard to X^*, we can match tokens between X^* and X using word embedding similarity as follows:

$$d(\mathcal{F}, \mathcal{Y}) = \left(\sum_{u \in X^*} \min_{v \in X}(dist(u, v)) \right) / |X^*|$$

where $dist(u, v) \geq 0$ is the semantic distance between two words u and v estimated from their pre-computed word embedding vectors [Pennington et al., 2014]. We use cosine distance in the Qsearch implementation, re-scaled for normalization to [0,1]. In the above equation, we map each word of query context X^* to its closest word in the fact context X in the embedding space. This scoring formula gives the same weight to every token in the query context X^*, which might be misleading, since they could have a different degree of importance. This issue is overcome by giving higher weight to important words and lower weight to uninformative words, using the following distance function:

$$d(\mathcal{F}, \mathcal{Y}) = \frac{\sum_{u \in X^*} W(u) \min_{v \in X}(dist(u, v))}{\sum_{u \in X^*} W(u)} + 1$$

where $W(u) \geq 0$ is the importance weight of word u. There are several weighting functions that can be used for W (e.g., *inverse document frequency (idf)*, *term strength*, etc.); we use Robertson's *idf* [Robertson, 2004]. We call the above formula the *directed embedding distance*, $ded(X^* \rightarrow X)$, between query and fact contexts.

$ded(X^* \rightarrow X)$ describes how well each word in X^* matches with some other word in X, but in many cases it fails to reflect the match between their meaning. The presence of a single word in the fact context X can totally change its meaning. Consider, as a concrete example, the two contexts $X^* = \{$*net, worth*$\}$ vs. $X = \{$*negative, net, worth*$\}$. Hence, our idea is to penalize the relevance score with an amount proportional to the directed embedding distance between X and X^*. Specifically, we define the *context embedding distance (ced)* that implements this idea:

$$d(\mathcal{F}, \mathcal{Y}) = ced(X^*, X) = ded(X^* \to X) \times ded(X \to X^*)^{\alpha}$$
$$= \left(\frac{\sum\limits_{u \in X^*} W(u) \min\limits_{v \in X}(dist(u, v))}{\sum\limits_{u \in X^*} W(u)} + 1 \right) \times \left(\frac{\sum\limits_{u \in X} W(u) \min\limits_{v \in X^*}(dist(u, v))}{\sum\limits_{u \in X} W(u)} + 1 \right)^{\alpha}$$

Intuitively, our *ced* measure is the product of two components: (1) $ded(X^* \to X)$ captures how well query context tokens match with fact context, and (2) $ded(X \to X^*)$ reflects how much additional terms in X shift its meaning, and hence, should be penalized. Parameter $\alpha \in [0, +\infty)$ controls how much the penalty scaling affects the total score.

Example 3.4.1. Consider the Qquery context $X^* = \{$*gross, domestic, product*$\}$ and two Qfact contexts $X_1 = \{$*gross, national, product*$\}$, $X_2 = \{$*gross, domestic, product, capita*$\}$. While we are more inclined to X_1 than X_2, the directed embedding distance $ded(X^* \to X_2)$ has a slightly better score than $ded(X^* \to X_1)$, as it does not penalize the word *"capita"* (which indicates that the GDP is per capita, not the total GDP). In contrast, $ded(X_1 \to X^*)$ is lower than $ded(X_2 \to X^*)$ (since *"national"* is close to *"domestic"*), preferring X_1 over X_2 with regard to X^*, which results in the desired ranking based on the context embedding distance *ced*. □

3.4.5 Entity Scoring

The output of Qsearch is a ranked list of entities from matching Qfacts with the Qquery. We assign a score for each candidate entity based on one of the above context distance models and aggregating over the entity's quantity-context pairs as follows:

$$score(c \in \mathcal{C}, \mathcal{Y}) = \min_{(q,X) \in L_c} d(\mathcal{F} = (c, q, X), \mathcal{Y})$$

where $d(\mathcal{F}, \mathcal{Y})$ is either the Kullback-Leibler divergence $KL(X^*, X)$ or the context embedding distance $ced(X^*, X)$. So when the same candidate entity appears in multiple Qfacts, we pick the best-scoring Qfact context distance.

3.5 Evaluation

We run experiments on a Linux machine with 80 CPU cores, 500GB RAM, and 2 GPUs. To evaluate Qsearch, we perform an intrinsic evaluation of our *Qfact extraction model* and an extrinsic evaluation of the *end-to-end Qsearch system*.

Dataset. All experiments use a large collection of news articles, compiled from two real world datasets: the *STICS* project [Hoffart et al., 2014] with news from 2014 to 2018, and the *New York Times* archive [Sandhaus, 2008] with news from 1986 to 2008. In total, our corpus consists of 7.6M documents.

3.5.1 Intrinsic Evaluation of the Quantity Fact Extraction Model

Training Setup. We implemented the LSTM network using Theano library, largely following [He et al., 2017] for the training configuration: using Adadelta with $\epsilon = 1e^6$ and $\rho = 0.95$; *lstm_hidden_unit = 300*; *rnn_dropout_prob = 0.1*; *batch_size = 100*.

We extracted training samples from the corpus using the distant-supervision technique as described in Section 3.3 and conducted the training process with different settings. In the *General* setting, we use all available training data of 3.2M training samples, where we maintain the ratio 3:1 between the number of positive and negative samples. We also train our model for three other *measure-specific* settings, where only a subset of the training samples is used. In particular, we classify training samples into different categories based on the quantity unit. For example, training samples containing quantities with unit *Kilometer* or *Meter* are chosen to train the model in the *Length* setting, while the ones with unit *US dollar*, *Euro*, etc. are picked for the *Money* setting. Among many such categories, we selected the three most prevalent measures *Money*, *Percentage* and *Length*, containing 307K, 235K and 41K training samples, respectively (also with ratio 3:1 between positive and negative samples). The trained models are then applied to the entire corpus to extract more Qfacts.

Performance of Extraction model. As the test data does not have any ground-truth labels, we randomly selected 100 samples that contain at least two entities from the output tag sequences, for each training model, and manually assessed their validity.

We evaluate the quality of the three output labels *<E>*, *<X>* and *<O>* by three measures: *Precision*, *Recall*, and *F1 score*. The results are shown in Table 3.1. We observe that all training models perform very well on entity tagging with more than

Tag	Length			Money			Percentage			General		
	Prec.	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*	*Prec.*	*Rec.*	*F1*
E	0.860	0.860	0.860	0.850	0.850	0.850	0.794	0.770	0.782	0.882	0.820	0.850
X	0.650	0.849	0.736	0.717	0.844	0.776	0.659	0.827	0.734	0.728	0.713	0.721
O	0.958	0.886	0.920	0.942	0.886	0.913	0.947	0.888	0.917	0.895	0.906	0.900
Mac.-avg.	0.823	0.865	0.839	0.836	0.860	0.846	0.800	0.828	0.811	0.835	0.813	0.824

Table 3.1: Evaluation of Qfact extraction model of Qsearch on different settings.

Domain	Distribution of queries based on unit of quantity				
	Money	*Length*	*Percentage*	*Others*	*Examples for Others*
Finance	76 %	-	12 %	12 %	no. of sales, albums, etc.
Transport	4 %	32 %	-	64 %	MPG, mph, horsepower, etc.
Sports	8 %	32 %	-	60 %	sec, years, kg, no. of medals, etc.
Technology	20 %	20 %	8 %	52 %	megapixels, Watt, mAh , etc.

Table 3.2: Statistics of benchmark queries from each domain.

85% *F1 score*. We also see that the measure-specific training variants for Length and Money have slight advantages.

3.5.2 Extrinsic Evaluation of the End-to-End Qsearch System

We performed the extrinsic evaluation of Qsearch on a benchmark of 100 quantity queries, collected by crowdsourcing and covering four domains: *Finance*, *Transport*, *Sports* and *Technology*. These queries capture a wide diversity of measures and units as well as variety in query formulations (e.g., phrases for the comparison operators); see Table 3.2. Anecdotal examples of user queries and their answers produced by Qsearch are shown in Table 3.3. We also considered queries from the QALD-6-task-3 statistical QA benchmark [Unger et al., 2016], but out of total 150 training and test queries, we found only 6 with quantity conditions (as opposed to simpler property lookups).

In this evaluation, we use the Qfact extraction model trained under the *General* setting, as it generalizes to different measures and units.

Setup. For each Qquery, we consider top-10 results returned by Qsearch and evaluate their relevance and validity by judgements from crowd-workers (using Figure-Eight platform, formerly known as CrowdFlower). The judges were shown the query, the top-10 entity answers, and the corresponding 10 sentences from which the answers were extracted. Each result was annotated as *relevant* or *irrelevant* to the query based on the cue given in its corresponding sentence. For each query, we collected three judgements

Domain	Query
Finance	**Q1:** Coal companies with more than 200 Million dollar annual profit
Transport	**Q2:** Sport utility vehicles with engine power at least 150 horsepower
Sports	**Q3:** Sprinters who ran 100 meter in less than 10 seconds

Query	Result	Corresponding Sentence
Q1	Duke Energy	Duke Energy had revenue of $ 23.9 billion and profit of $ 1.9 billion last year.
Q2	Ford Escape	Its V-6 engine (the Escape is a four-cylinder) has 270 horsepower, 20 percent more than the Lexus RX330.
Q3	Andre Grasse	Andre De Grasse, a 20-year-old from Markham, Ont., has run the 100 metre in under 10 seconds three times this year.
Q4	Nikon D7100	For example, the D7100 can be found in a kit with 18-140 mm and 55-300 mm lenses , so you'll want to use the 55-300 mm and zoom in to 300 mm.

Table 3.3: Anecdotal examples of quantity queries and results from Qsearch.

and used the majority label as gold standard. Overall, we obtained a high inter-annotator agreement with Fleiss' Kappa value of 0.54.

Baselines. Although our Qsearch system produces entities as main result, we still want to compare it with standard search systems, which produce snippets. As there is no other system that can handle quantity queries with crisp entity answers, we use search systems as baselines that produce text snippets as answers. Specifically, we ran all benchmark queries on Elasticsearch, locally indexing all sentences of our news corpus, and on Google web search retrieving the top-10 result snippets. Elasticsearch uses a text-oriented state-of-the-art ranking model based on BM25.

The baselines were given certain advantages, to avoid that Qsearch could be viewed as an unfair competitor. For Elasticsearch, we consider only sentences that contain an entity and a quantity. For the evaluation, we asked crowd-workers to annotate top-10 results, retrieved from Elasticsearch, as relevant or irrelevant based on whether they spotted a reasonable result for the quantity query. To evaluate result snippets from Google search, we instructed annotators to be generous, as the result snippets are not well-formed sentences (but could be synthesized from non-contiguous text segments with ellipses). For example, a text snippet that contains a correct entity and its quantity is considered relevant even if it also contains other entities or quantities. Such instructions to annotators give Google results an advantage because Qsearch results are considered relevant only if both entity and quantity are correctly extracted.

We also explored several state-of-the-art QA systems over linked open data: Frankenstein [Singh et al., 2018], QAnswer [Diefenbach et al., 2019], Platypus [Tanon et al.,

Metric	**Finance**		**Transport**		**Sports**		**Technology**		**All**	
	KL-div.	*Emb.*	*KL-div.*	*Emb.*	*KL-div.*	*Emb.*	*KL-div.*	*Emb.*	*KL-div.*	*Emb.*
Pr.@1	0.720	0.800	0.480	0.600	0.560	0.680	0.640	0.680	0.600	0.690
Pr.@3	0.667	0.747	0.480	0.480	0.507	0.587	0.627	0.653	0.570	0.617
Pr.@5	0.632	0.672	0.412	0.412	0.480	0.528	0.550	0.624	0.519	0.559
Pr.@10	0.604	0.608	0.333	0.379	0.412	0.432	0.500	0.547	0.462	0.492
Hit@3	0.880	0.920	0.760	0.760	0.760	0.800	0.840	0.880	0.810	0.840
Hit@5	0.880	0.960	0.760	0.760	0.920	0.840	0.840	0.920	0.850	0.870
MRR	0.792	0.870	0.621	0.678	0.685	0.746	0.747	0.783	0.711	0.769

Table 3.4: End-to-end evaluation of Qsearch.

2018], AskNow [Dubey et al., 2016], Quint [Abujabal et al., 2017], SPARKLIS [Ferré, 2017]. None of these systems is geared for handling quantity questions, except SPARKLIS, however it can only process quantities without associated unit. Moreover, their underlying KBs have poor coverage of quantities. They failed on almost all of our benchmark queries; so we excluded these systems from our comparative evaluation.

Performance of Qsearch. Table 3.4 shows the performance of Qsearch for the four domains and for all 100 queries together, using the two variants of our ranking models: KL divergence and context embedding distance (*ced*). For *ced* we empirically tune the parameter $\alpha = 3$ based on results from 10 validation queries disjoint from the 100 test queries. We report three metrics: *Precision@k*, *Hit@k* and *Mean-Reciprocal-Rank (MRR)*, macro-averaged over queries. We do not discuss metrics like Recall or MAP, as these would require exhaustively annotating a huge pool of candidate answers.

Overall, Qsearch performs amazingly well, typically with MRR around 0.7 or better. The best results are for the Finance domain, which has the highest share in the corpus and is most represented in the Qfact extraction training. *Precision@1* is pretty good, but precision drops substantially when going deeper in the rankings. The embedding-based ranking model clearly outperformed the KL-divergence method by a significant margin.

Figure 3.3 presents the comparison of Qsearch with the *ced* ranking model against Elasticsearch and Google, showing the metrics *Prec.@3*, *Prec.@5*, *Hit@3* and *MRR*. The results clearly indicate that Qsearch outperforms both baselines by a large margin.

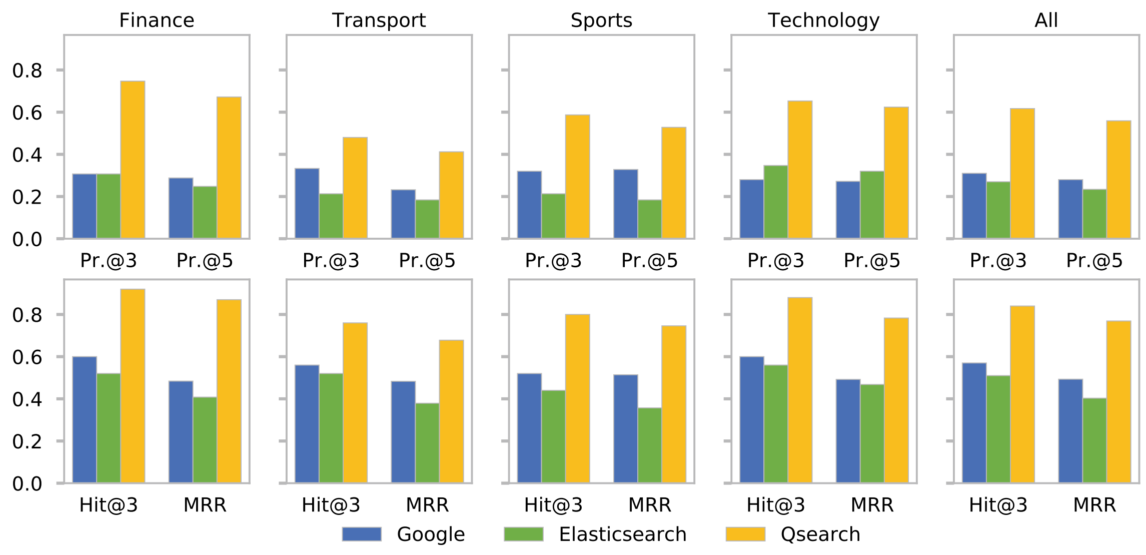
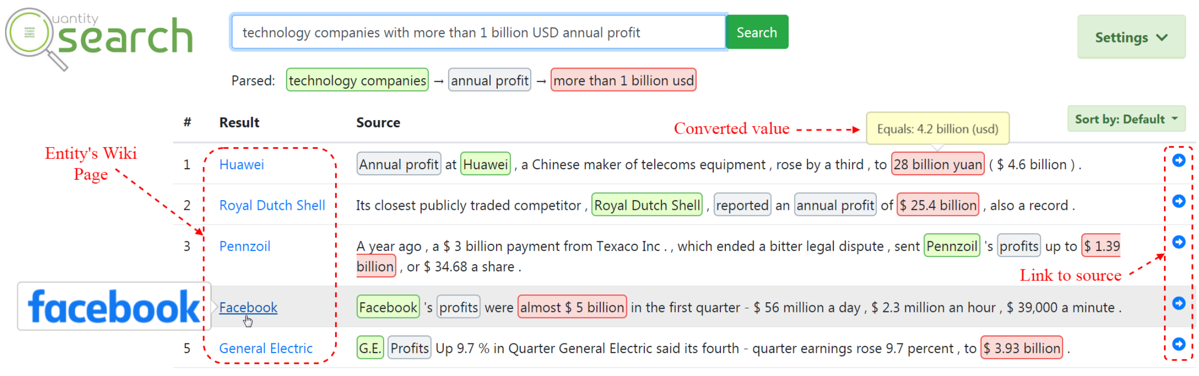


Figure 3.3: Comparison of Qsearch against baselines.



technology companies with more than 1 billion USD annual profit

Parsed: technology companies → annual profit → more than 1 billion usd

Equals: 4.2 billion (usd)

#	Result	Source
1	Huawei	Annual profit at Huawei , a Chinese maker of telecoms equipment , rose by a third , to 28 billion yuan ($ 4.6 billion) .
2	Royal Dutch Shell	Its closest publicly traded competitor , Royal Dutch Shell , reported an annual profit of $ 25.4 billion , also a record .
3	Pennzoil	A year ago , a $ 3 billion payment from Texaco Inc . , which ended a bitter legal dispute , sent Pennzoil 's profits up to $ 1.39 billion , or $ 34.68 a share .
4	Facebook	Facebook 's profits were almost $ 5 billion in the first quarter - $ 56 million a day , $ 2.3 million an hour , $ 39,000 a minute .
5	General Electric	G.E. Profits Up 9.7 % in Quarter General Electric said its fourth - quarter earnings rose 9.7 percent , to $ 3.93 billion .

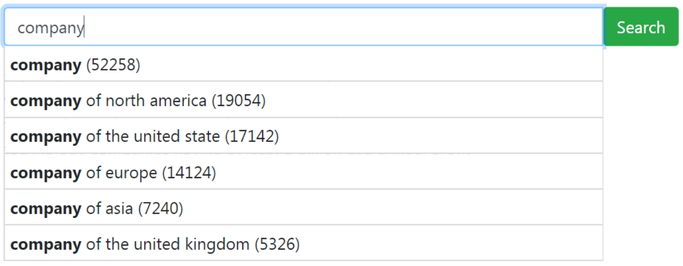
Figure 3.4: Qsearch Web interface.

3.6 Qsearch Demonstration

We developed a Web interface to allow users to experience our Qsearch system[2]. Figure 3.4 shows a screenshot of the top search results for an example quantity query.

Input. Users can provide query in natural language text as input to the system. To give better search experience, we guide users by showing a list of top YAGO types along with the number of relevant entities existing in our database while writing the query. Figure 3.5 illustrates this feature.

[2]https://qsearch.mpi-inf.mpg.de/



company (52258)
company of north america (19054)
company of the united state (17142)
company of europe (14124)
company of asia (7240)
company of the united kingdom (5326)

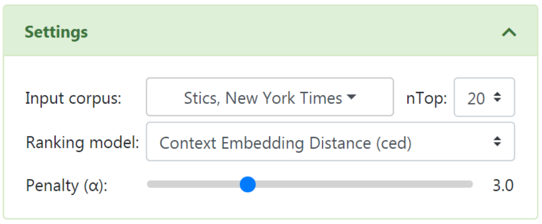
Figure 3.5: Type suggestion in Qsearch.

Output. Qsearch processes the input query and generates a ranked list of entities relevant to the input query. The parsed query, top answers and their entity pages (from Wikipedia) are shown to users, along with the text snippets from where the Qfacts are extracted, and the links to the original web pages (see Figure 4.2). For better experience, we also show the representation images of the entity answers (extracted from the entities' Wikipedia pages) to the users when they hover on the entity links. For each answer, Qsearch highlights the extracted entity, quantity and context cues of the Qfact, and also provides the converted quantity value if its unit is different from the one being asked in the query.

Exploration of underlying data sources. Qsearch answers quantity queries based on information from three large collections of text documents - two real-world news corpora, the *STICS* project [Hoffart et al., 2014] (containing 5.84M documents) and the *New York Times* archive [Sandhaus, 2008] (containing 1.77M documents), and a collection of web pages from the English Wikipedia[3](containing 5.78M documents). In total, our text data consists of 13.39M documents. By default, Qsearch uses all three collections to generate answers. However, we allow users to change this setting (see Figure 3.6) in order to narrow down their search on specific text collections. As the characteristic of news articles is quite different from Wikipedia articles, using both types of text collections as underlying input data allows Qsearch to answer a wide range of quantity queries. For example, information about finance or sport domain can be found all over the news articles and also in Wikipedia, but answers for queries on geological objects (e.g., glaciers with length more than 100 km) are mainly covered only by Wikipedia articles.

Exploration of answer generation methods. Qsearch employs different ranking models as mentioned in Section 3.4 to rank the entity answers and shows top confident results to the users. In the default configuration, Qsearch uses the *context embedding distance*

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


Settings

Input corpus: Stics, New York Times

nTop: 20

Ranking model: Context Embedding Distance (ced)

Penalty (α): 3.0

Figure 3.6: Qsearch options.

(ced) as the ranking model, and shows top 20 answers. We provide a flexible interaction to the system by allowing users to modify these search settings (as shown in Figure 3.6) to explore the answer generation methods. Here, users can adjust the number of retrieved results (10, 20, 30, 40 or 50) or change the underlying ranking model (*context embedding distance* or *KL-divergence*) and its parameter. Additionally, Qsearch provides further exploration with a secondary ranking criterion, where top retrieved answers can be re-ranked either by quantity value (after normalization), or by entity prominence (based on view count of entities' Wiki pages). This secondary ranking criterion can be set using *"Sort by"* button in Figure 3.4.

3.7 Related Work

Question Answering. QA over knowledge graphs and other linked data sources has received great attention over the last years; see [Unger et al., 2014, Diefenbach et al., 2018] for surveys. State-of-the-art methods (e.g., [Yih et al., 2015, Bast and Haussmann, 2015, Xu et al., 2016, Abujabal et al., 2018, Zheng et al., 2018]) translate questions into SPARQL queries, bridging the gap between question vocabulary and the terminology of the underlying data by means of templates and/or learning from training collections of question-answer pairs. Benchmarks like the long-standing QALD series and other competitions have shown great advances along these lines [Usbeck et al., 2019]. However, these benchmark tasks hardly contain any quantity queries of the kind addressed here (even in QALD-6-task-3, only 6 out of 150 questions are of this kind, others are mostly about quantity lookup). Note that look-ups of quantity attributes of qualifying entities (e.g., Jeff Bezos's net worth, 10 richest people, or fastest sprinter over 100m) are of a different nature, as they do not contain quantity comparisons between query and data (e.g., worth more than 50 million USD, running faster than 9.9 seconds). Moreover, the scope and diversity of the benchmark queries is necessarily restricted to relatively few numeric properties, as knowledge graphs hardly capture quantities in their full extent (with value and unit properly separated and normalized). This is our motivation to tap into text sources with more extensive coverage.

QA over text has considered a wide range of question types (e.g., [Yang et al., 2018, Clark and Gardner, 2018, Chen et al., 2017]), but there is again hardly any awareness of quantity queries. Keyword search, including telegraphic queries, with quantity conditions have been considered by [Joshi et al., 2014], and have been applied to web tables [Pimplikar and Sarawagi, 2012, Sarawagi and Chakrabarti, 2014].

[Banerjee et al., 2009] and its follow-up work [Sarawagi and Chakrabarti, 2014] focused on a specific kind of quantity query, namely, retrieving and aggregating numerical values associated with an attribute of a given entity (e.g., Bezos's net worth or GDP of India). To this end, learning-to-rank techniques over value distributions were developed to counter the uncertainty in the retrieved values, where web pages often contain crude estimates and lack exact values. In contrast to our setting, that work did not consider quantities in search conditions.

Information Extraction. Recognizing and extracting numeric expressions from text has been addressed using techniques like CRFs and LSTMs (e.g., [Madaan et al., 2016, Saha et al., 2017, Alonso and Sellam, 2018]). However, this alone does not turn numbers into interpretable quantities, with units and proper reference to the entity with that quantity. Only few works attempted to canonicalize quantities by mappings to handcrafted knowledge bases of measures [Ibrahim et al., 2016], but these efforts are very limited in scope. The special case of temporal expressions has received substantial attention (e.g., [Strötgen and Gertz, 2016]), but this solely covers dates as measures.

Most related to our approach are the works of [Sarawagi and Chakrabarti, 2014] and [Roy et al., 2015]. The former used probabilistic context-free grammars to infer units of quantities, but focused specifically on web tables as inputs. The latter extended semantic role labeling (see below) to extract quantities and their units from natural language sentences. Neither of these can be readily applied to extracting quantities and their reference entities from arbitrary textual inputs.

Semantic Role Labeling. Semantic role labeling (SRL) has been intensively researched as a building block for many NLP tasks [Gildea and Jurafsky, 2002]. Given a verb phrase of a sentence viewed as a central predicate, SRL identifies phrases that are assigned to pre-defined roles to form a frame-like predicate-arguments structure. Modern SRL methods make use of pre-computed word embeddings and employ deep neural networks for role filling (e.g., [Zhou and Xu, 2015, He et al., 2017, FitzGerald et al., 2015]). Our approach differs from this state-of-the-art SRL, as we are not primarily focused on the verb-phrase predicate, but consider the numeric quantity in a sentence as the pivot and aim to capture quantity-specific roles.

To support exploration of quantitative facts in financial reports, [Lamm et al., 2018] proposed a semantic representation for quantity-specific roles. [Roy et al., 2015] devised a quantity representation as an additional component of an SRL method, which is part of the Illinois Curator software suite. Our approach makes use of this technique, as a preprocessing step. However, we go further by learning how to connect quantities with their respective entities and to collect relevant context cues that enable our matching and ranking stage for query answering.

3.8 Summary

Awareness of entities and types has greatly advanced semantic search both for querying the web of linked data and for Internet search engines. In contrast, coping with quantities in text content and in query constraints has hardly received any attention, yet is an important case. This chapter has presented the Qsearch system for full-fledged support of quantity queries, through new ways of information extraction and answer matching and ranking. We capture quantities in their full extent, including units of measures, reference entities and the relevant contexts. The model for Qfacts and Qqueries is relatively simple but highly versatile and effective. A key asset of Qsearch is its high quality in extracting Qfacts, recognizing the right entity-quantity pairs even in complex sentences.

Chapter 4

QuTE: Extracting Contextualized Quantity Facts from Web Tables

4.1 Introduction

4.1.1 Motivation and Problem

The core of answering quantity-filter queries is the problem of extracting entity-quantity facts from web sources. In the previous chapter, we have addressed this problem 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 [Sarawagi and Chakrabarti, 2014, Roy et al., 2015, Saha et al., 2017]. In this chapter, 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, for the purpose of answering quantity queries. An illustrative example, which could serve to answer the query about British football teams, is shown in Table 4.1.

There are good prior works on extracting entity-centric facts from web tables, including surveys [Cafarella et al., 2018, Dong et al., 2020a, Zhang and Balog, 2020]. 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 [Shen et al., 2015], 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 4.1 are just strings, varying in units and scale and

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

Table 4.1: Illustrative example on football teams.

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 4.1, 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 [Cafarella et al., 2018, Zhang and Balog, 2020]. 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.

4.1.2 Approach

This chapter 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 [Yagnik and Islam, 2007].

4.1.3 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 4.3 and 4.4).

- 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 4.4).

- We present a new method for corroborating extracted facts at query time, reranking them and pruning false positives based on a technique for consistency learning (Section 4.5).

- 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 [Ho et al., 2019] and a new collection of 150 queries with list-based ground-truth.

We make the experimental data and code available to the research community[1].

4.2 Model and System Overview

4.2.1 Model

The *input* for fact extraction is a collection of ad-hoc tables, from a web crawl, spreadsheet corpus or Wikipedia dump (e.g., [Eberius et al., 2015]).

[1]https://www.mpi-inf.mpg.de/research/quantity-search/quantity-table-extraction

Definition 4.2.1 (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 [Cafarella et al., 2018].

Definition 4.2.2 (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.*

Example 4.2.1. In Table 4.1, 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. [Ho et al., 2019] where this terminology is defined for text-based extraction).

Definition 4.2.3 (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;
- 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).

Example 4.2.2. A perfect extractor from Table 4.1 should produce Qfacts such as (*Estadio Santiago Bernabéu, 81044, "stadium, capacity, seats, Madrid"*), (*Chelsea F.C.,*

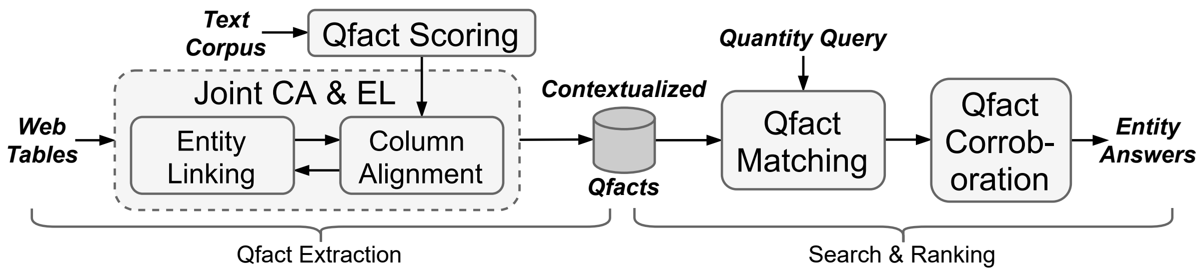


Figure 4.1: Overview of the QuTE system.

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 [Ho et al., 2019]:

Definition 4.2.4 (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 4.2.5 (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).

4.2.2 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 4.1.

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 [Sarawagi and Chakrabarti, 2014] and Illinois Quantifier [Roy et al., 2015]. 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 [Sarawagi and Chakrabarti, 2014] for normalization, including unit conversions.

For **entity recognition**, we employ the AIDA dictionary[2], 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 [Limaye et al., 2010, Bhagavatula et al., 2015, Ibrahim et al., 2016, Efthymiou et al., 2017, Ritze and Bizer, 2017]. We follow [Bhagavatula et al., 2015], 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.

4.3 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 4.3.1 (Column Alignment). *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:* $\Lambda = \{C_{k_i} \rightarrow C_{v_j} |\ i \in \{1..x\}\}$

[2] https://github.com/ambiverse-nlu

4.3.1 Heuristics and their Limitations

Column alignment has been addressed in prior works [Venetis et al., 2011, Deng et al., 2013, Braunschweig et al., 2015] 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 4.3.2 (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 4.3.3 (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 4.3.4 (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 4.1, 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., [Limaye et al., 2010, Bhagavatula et al., 2015, Ibrahim et al., 2016, Efthymiou et al., 2017, Ritze and Bizer, 2017]), 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 [Braunschweig et al., 2015]. 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 4.1 is a typical case, and the method of [Braunschweig et al., 2015] does not add any benefit over the simpler heuristics here. Hence we disregard this method.

4.3.2 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 4.3.5 (CA-score). *The quality of a column alignment* Λ *is:*

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

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 4.4.

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 4.3.6 (Plausibility Maximization). *We define the plausibility of interpreting table T with entity linking Φ and column alignment Λ as:*

$$\lambda \cdot \textit{CA-score}(\Lambda|\Phi) \;+\; (1-\lambda) \cdot \textit{EL-score}(\Phi|\Lambda) \tag{4.1}$$

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 [Lu and Getoor, 2003], called *ICA*, which was also used by [Bhagavatula et al., 2015]. 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 4.1) is chosen as the final result.

4.3.4 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 4.1. 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.

4.4 Qfact Confidence Scoring

This section explains how we utilize external text corpora to compute $\textit{ext-score}(\mathcal{F})$ for the CA-score model of Section 4.3.2.

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 [Ho et al., 2019]: 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[3] 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 4.4.1 (Evidence Score). *For Qfact $\mathcal{F} = (e, q, X = h_k)$, the evidence score from collection $\mathbb{C}$ is:*

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

$$\textit{where} \quad \textit{sim}\big((q, X), (q', X')\big) = w_1 \cdot \textit{sim}_1(q, q') + w_2 \cdot \textit{sim}_2(X, X')$$

with tunable coefficients w_1, w_2.

The function finds the best matching evidence Qfact in $\mathbb{C}$ with same entity e. $\textit{sim}_1$ 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. $\textit{sim}_2$ compares the Q-column header $X = h_k$ with the evidence context bag-of-words X' by the *directed embedding distance* of [Ho et al., 2019]. This rewards if the column name appears in X', but also gives credit to different words that are related by their word2vec embeddings.

[3]https://github.com/huggingface/neuralcoref

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 4.4.2 (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:*

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

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 [Wu and Palmer, 1994] 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

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.

Table 4.2: Top-scoring text evidence for Qfact candidate (*Allianz Arena, 75000, "Capacity"*).

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

4.5 Use Case: Quantity Querying

4.5.1 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 $\mathcal{Q} = (qt, qq, qX)$ against this data by mostly following the method of [Ho et al., 2019]: 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 [Ho et al., 2019]. This yields a ranking of entity answers to a given query.

We extend the context comparison, as our setting differs from [Ho et al., 2019]. 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 4.5.1 (Weighted Directed Embedding Distance).

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

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 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 $\sigma(v)$.

4.5.2 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 [Yagnik and Islam, 2007]. 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-ded* $-$ *cons-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.

```
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 $\mathcal{Q} = (qt, qq, qX)$
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.
```

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*:

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

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.).

4.6 Evaluation

4.6.1 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 4.6.2.

Hyper-Parameter Tuning. Our method has a number of hyper-parameters for Qfact extraction: λ in Equation 4.1; 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 [Bhagavatula et al., 2015], 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 [Ibrahim et al., 2016]. 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 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 4.3.1): *(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 [Venetis et al., 2011].

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 [Venetis et al., 2011]	0.864	0.717	0.597
Iterative CA	0.934	0.900	0.769

Table 4.3: Column alignment precision *(macro_avg)*.

Method	*Wiki_L-R_Qt*	*Equity_Qt*	*Wiki_Diff*
Prior	0.849	0.821	0.846
EL-MRF [Bhagavatula et al., 2015]	0.893	0.863	0.902
Joint EL&CA	0.900	0.876	0.902

Table 4.4: Entity linking precision *(micro_avg)*.

The results are shown in Table 4.6.1. 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 [Bhagavatula et al., 2015, Ibrahim et al., 2016], 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* [Bhagavatula et al., 2015] 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 4.6.1 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 4.6.1.

We observe that without type-lifted evidences, the CA precision decreases by more

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 4.5: Ablation study results for CA.

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.

4.6.2 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 4.6.1, we tune query-time hyper-parameters for the *w-ded* distance and for the mixture with *cons-score* (see Section 4.5.2) 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 [Ibrahim et al., 2019]. 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 [Bhagavatula et al., 2015]. 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 4.6.2 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

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

Table 4.6: Table collection statistics.

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 [Ho et al., 2019], 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 [Ho et al., 2019]. 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:

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 4.7: Performance results for *Q100*.

- *Qsearch*[4] is a text-based quantity search engine from our prior work [Ho et al., 2019, Ho et al., 2020]. 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.

[4]https://qsearch.mpi-inf.mpg.de/

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

Table 4.8: Performance results for *NewQ150*.

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

Main Results. The precision results for the *Q100* benchmark are shown in Table 4.6.2. 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”.

The results for the *NewQ150* benchmark are shown in Table 4.8, 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 4.9 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

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 4.10: Ablation study results on *Q100*.

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

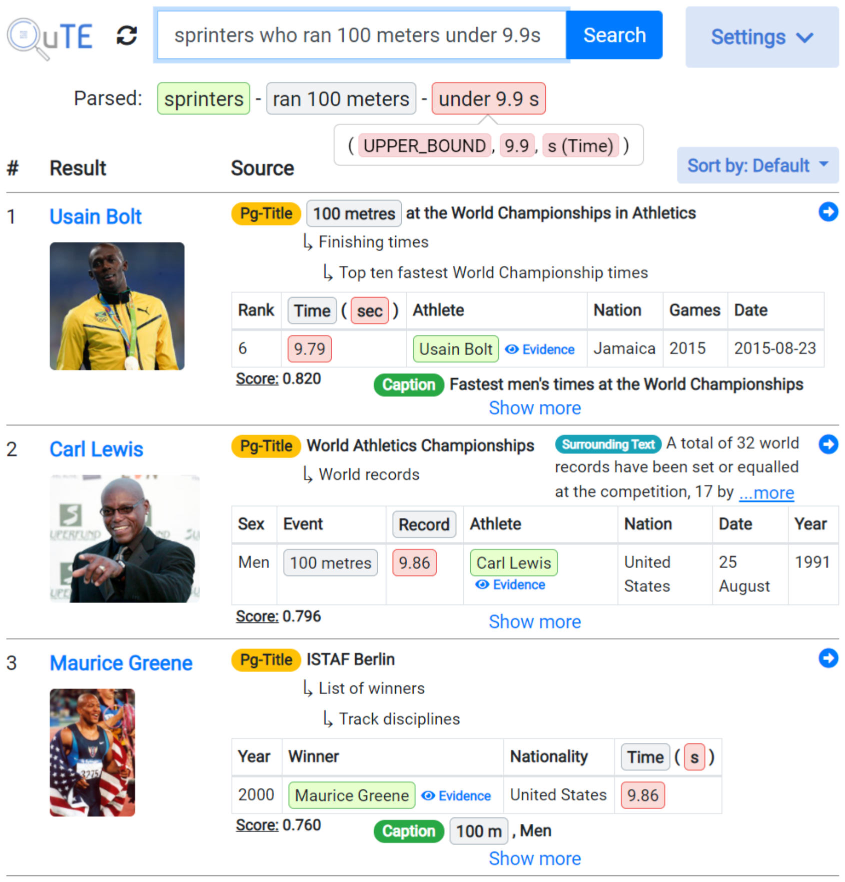
Table 4.11: Ablation study results on *NewQ150*.

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 4.6.2 and 4.11 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 4.6.2 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



QuTE sprinters who ran 100 meters under 9.9s Search Settings

Parsed: sprinters - ran 100 meters - under 9.9 s

(UPPER_BOUND , 9.9 , s (Time))

# Result Source Sort by: Default

1 Usain Bolt

Pg-Title 100 metres at the World Championships in Athletics
↳ Finishing times
↳ Top ten fastest World Championship times

Rank	Time (sec)	Athlete	Nation	Games	Date
6	9.79	Usain Bolt Evidence	Jamaica	2015	2015-08-23

Score: 0.820 Caption Fastest men's times at the World Championships
Show more

2 Carl Lewis

Pg-Title World Athletics Championships
↳ World records

Surrounding Text A total of 32 world records have been set or equalled at the competition, 17 by ...more

Sex	Event	Record	Athlete	Nation	Date	Year
Men	100 metres	9.86	Carl Lewis Evidence	United States	25 August	1991

Score: 0.796
Show more

3 Maurice Greene

Pg-Title ISTAF Berlin
↳ List of winners
↳ Track disciplines

Year	Winner	Nationality	Time (s)
2000	Maurice Greene Evidence	United States	9.86

Score: 0.760 Caption 100 m , Men
Show more

Figure 4.2: QuTE query and top-ranked answers.

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

We illustrate the complementarity of text and tables by a challenging example: *"airplanes with more than 12000 kilometers range"*. Qsearch returns 7 answers, out of which 5 are correct (2 relate to routes involving stops); while the table-based system QuTE finds a table *"flight distance records"*, which although not providing full coverage, contains 9 correct results. Only one answer (Boeing-787) is shared among the two result sets, while all other results are unique to their paradigms.

4.7 QuTE Demonstration

QuTE is accessible through a web interface[5]. Figure 4.2 shows a screenshot of top-ranked results for the example query "sprinters who ran 100 meters under 9.9s".

Data Sources. QuTE runs on a repository of ca. 18M Qfacts for ca. 800K entities, extracted from two large corpora: *WikiTables* consisting of 1.8M tables from English Wikipedia, and *TableL* [Ibrahim et al., 2019] consisting of 2.6M tables from 1.5M Common Crawl web pages. The text-based witnesses draw from a text corpus of ca. 13.5M documents from Wikipedia and news articles.

Input. QuTE accepts user input in two modes: form-based and free-text. In form-based mode, users enter a query into three text fields *(entity-type, quantity-filter, context-cues)*, with auto-completion suggestions for types. This mode helps our engine to evaluate the query as precisely as possible. In free-text mode, user queries are telegraphic phrases or full-fledged questions, which are then decomposed into the three constituents by a rule-based parser, as shown in the demo. The parser uses the Yago type taxonomy and a dictionary of quantity units.

To switch between the two input modes, users click on the toggle button on the left side of the search bar.

Output. QuTE computes a ranked list of entity answers, based on their confidence scores (see Section 4.5). The parsed query and top-k answers are shown to the users, along with context cues for each answer. Different colors are used to highlight answer entity (in light green), answer quantity (in red), column headers (obvious), page title (in orange), DOM-tree headings (in orange), table caption (in green), table row (obvious) and a surrounding-text snippet (in cyan), when available. By default, QuTE shows only the highest-scoring Qfact for each distinct entity. By clicking on the *"Show more"* button, additional Qfacts for the same entity can be displayed. These often include a noisy tail, showing the difficulty of the task.

Table 4.12 shows more anecdotal examples of quantity queries and their top answers produced by QuTE. For comparison, the table also shows top-3 results from a major search engine (SE), with incorrect, uninformative or very partial results in purple. These are evidence that search engines do not properly interpret quantity queries and simply return string-level matches. Some are high-quality list pages: browsing through them would give the user good results, but this requires extra effort. Moreover, even when

[5]https://qsearch.mpi-inf.mpg.de/table/

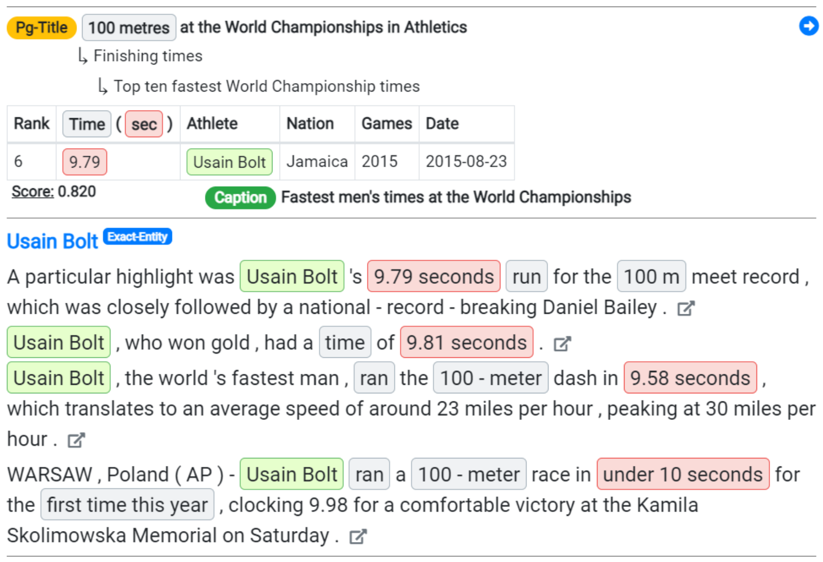
Query:	Buildings higher than 500 meters
QuTE	Burj Khalifa, One World Trade Center, Taipei 101, etc.
SE	List of tallest structures – 400 to 500 metres - Wikipedia, List of tallest structures - Wikipedia, China bans skyscrapers taller than 500 meters - Global Times
Query:	Which universities have more than 60000 students?
QuTE	Liberty University, University of Belgrade, University of Toronto, etc.
SE	List of United States public university campuses by enrollment, The top 50 US colleges that pay off the most 2019, There Are Now 50 Colleges That Charge More Than $60000
Query:	Stadiums with more than 100000 seats
QuTE	Melbourne Cricket Ground, Rungrado 1st of May Stadium, Tiger Stadium (LSU), etc.
SE	List of stadiums by capacity - Wikipedia, List of European stadiums by capacity - Wikipedia, World's Largest Sport Stadiums - Topend Sports

Table 4.12: Queries with top-3 answers by QuTE vs. search engine (as of March 5, 2021). Purple answers are uninformative or very partial.

top-ranked results are useful, there is often a bias towards specific selections (e.g., US colleges although the query intent is world-wide).

Exploring Contextual Cues. Matching query cues in Qfact contexts is a vital element and novelty of QuTE. Consider the example shown in Figure 4.2. If all matching would be based on table rows alone, we could merely return *Carl Lewis* as a query result. If the matching were further restricted to the entity-quantity pairs alone (as it was pursued in prior works on knowledge extraction from web tables), even that result would be missing. The other two results, on ranks 1 and 3, would be completely out of scope. It is only by cues from page title, table caption, etc. that QuTE is able to discover these answers and rank them highly. Note that this entails identifying the beneficial cues in the Qfact context, among many uninformative or query-irrelevant ones.

Exploring Text-based Evidence. The witnesses from the large text corpus are often interesting for users to obtain explanations and additional information. To showcase both the internal workings of QuTE and to support users in answer exploration, the demo allows viewing the witnesses by clicking on the *"Evidence"* button next to each answer entity. Figure 4.3 shows this for the answer *Usain Bolt* for the example query of Figure 4.2. The screenshot shows evidence snippets for the answer entity that match



Pg-Title 100 metres at the World Championships in Athletics

↳ Finishing times

↳ Top ten fastest World Championship times

Rank	Time (sec)	Athlete	Nation	Games	Date
6	9.79	Usain Bolt	Jamaica	2015	2015-08-23

Score: 0.820 Caption Fastest men's times at the World Championships

- Usain Bolt Exact-Entity
  - A particular highlight was Usain Bolt 's 9.79 seconds run for the 100 m meet record , which was closely followed by a national - record - breaking Daniel Bailey .
  - Usain Bolt , who won gold , had a time of 9.81 seconds .
  - Usain Bolt , the world 's fastest man , ran the 100 - meter dash in 9.58 seconds , which translates to an average speed of around 23 miles per hour , peaking at 30 miles per hour .
  - WARSAW , Poland (AP) - Usain Bolt ran a 100 - meter race in under 10 seconds for the first time this year , clocking 9.98 for a comfortable victory at the Kamila Skolimowska Memorial on Saturday .

Figure 4.3: Evidence for the Qfact answer about Usain Bolt.

the query structure. We highlight the entities, the quantities, and the matching context words in the text evidence, e.g., Usain Bolt, 9.79 seconds, {time, run, ran, 100 metres, . . . }.

Exploration of Different Configurations. Users can customize the search through a settings panel. QuTE can answer queries using either *WikiTables* or *TableL* or both corpora together (default). Further options are to customize the settings for column alignment and for consistency-based corroboration (see Sections 4.3.2 and 4.5.2).

By default, QuTE answers are ranked by confidence score. Users can re-rank them in ascending or descending order of quantity value, or by entity prominence via clicking on the *"Sort by"* button.

4.8 Related Work

Quantity Recognition. Detecting quantities in text and tables has been well researched, with prevalent methods based on rules, CRFs or neural learning [Sarawagi and Chakrabarti, 2014, Roy et al., 2015, Madaan et al., 2016, Alonso and Sellam, 2018, Saha et al., 2017, Ibrahim et al., 2016]. 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 [Cafarella et al., 2018] and [Zhang and Balog, 2020] discuss the relevant literature.

Our work builds on state-of-the-art entity linking for web tables [Limaye et al., 2010, Bhagavatula et al., 2015, Ibrahim et al., 2016, Efthymiou et al., 2017, Ritze and Bizer, 2017] sharing the general approach of niccombining 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 [Dong et al., 2014, Ritze et al., 2015, Ritze and Bizer, 2017, Oulabi and Bizer, 2019, Fetahu et al., 2019, Kruit et al., 2019]. 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 [Venetis et al., 2011, Deng et al., 2013]. 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 [Braunschweig et al., 2015]. 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 4.1 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 [Balog, 2018, Diefenbach et al., 2018, Huang et al., 2020, Reinanda et al., 2020]. 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 [Unger et al., 2015], NaturalQuestions [Kwiatkowski et al., 2019], ComplexWebQuestions [Talmor and Berant, 2018], LC-QuAD [Dubey et al., 2019] 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 [Ho et al., 2019, Ho et al., 2020], which was solely based on textual contents, though.

Search and QA over web tables have been addressed in various settings. Methods in [Pimplikar and Sarawagi, 2012, Sarawagi and Chakrabarti, 2014, Yakout et al., 2012, Chakrabarti et al., 2020] 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.

4.9 Summary

This chapter 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.

Chapter 5

QL: Enhancing Knowledge Bases with Quantity Facts

5.1 Introduction

5.1.1 Motivation and Problem

Large knowledge bases (KBs) [Hogan et al., 2021, Weikum et al., 2021], such as Wikidata [Vrandecic and Krötzsch, 2014], DBpedia [Lehmann et al., 2015], YAGO [Suchanek et al., 2007] or NELL [Carlson et al., 2010], contain many millions of entities and more than a billion of facts about them. The facts are typically represented in the form of subject-predicate-object (SPO) triples (along with qualifiers for higher-arity relations), and include, for example, properties of the Eiffel Tower like its location, architect, opening date, material, height, mass, and more.

The last two properties are examples of *quantity facts*: physical, technological or financial measures of interest, with proper units, associated with an entity. These are covered well for prominent entities, but KBs exhibit large gaps in quantities for less popular entities. For example, Wikidata knows more than 2 million buildings but has height entries for only less than 0.4% of these. For its 6183 sprinters, it has 100 meters race times, typically personal records, for only 38. For athletes running 400 meter hurdles, only 21 have information about their race times. Even Karsten Warholm, the Olympic champion and world record holder, falls into this huge gap[1].

The problem that we address in this chapter is how to close this gap in KB coverage. To this end, we present new methods for automatically extracting quantity facts from web contents, so as to augment the KB with the missing properties.

[1]see https://www.wikidata.org/wiki/Q13900927 as of Oct 12, 2021

5.1.2 State of the Art and Challenges

Information extraction (IE) from web contents like tables, lists and texts, has been greatly advanced over the past two decades [Smirnova and Cudré-Mauroux, 2019, Martínez-Rodríguez et al., 2020, Han et al., 2020]. However, it mostly focuses on identifying and classifying pairs of related entities for a given set of relations, typically based on patterns or distant supervision. There is not much work on extracting pairs of entities and quantities (other than for Wikipedia infoboxes and similarly semi-structured content). IE for quantity facts from text comes with the extra challenges of i) capturing the numeric value and proper unit as well as possible scaling factor from natural language, and ii) correctly connecting a quantity appearance to its entity mention in the same passage (not necessarily the same sentence).

These challenges have been studied in some prior works, most notably [Madaan et al., 2016, Roy et al., 2015, Saha et al., 2017, Sarawagi and Chakrabarti, 2014]. However, while these methods address the first challenge, they do not solve the second one: their outputs are OpenIE-flavored with strings as arguments of SPO triples not properly mapped onto KB entities. Moreover, they do not scale to large input corpora.

The closest works to ours are the QEWT [Sarawagi and Chakrabarti, 2014] and Qsearch [Ho et al., 2019] systems. QEWT taps into web tables as input and does not carry over to text input. Qsearch operates over large text corpora and satisfies the functional desiderata towards enhancing KB entities with quantities. However, both QEWT and Qsearch are geared for query answering rather than KB population. To this end, both optimize precision for top-ranked results of quantity-centric queries. Neither of them addresses the challenge of achieving high coverage of quantity facts for a given KB at large scale.

5.1.3 Approach and Contribution

To overcome the limitations of prior works, we present a novel method, called QL (for **Q**uantity **L**eap), for KB population with quantity facts. As input, our method takes a text corpus, a KB, a quantity relation to be populated (e.g., *height*) with its active domain given by a set of KB entities (e.g., *Eiffel_Tower*, *Empire_State_Building*, etc.), as well as the target units (e.g., *meter*, *feet*, *kilometer*) from the KB schema. Specifically, we use Wikidata as a KB, and produce output in the form of properly normalized quantity facts that can enhance the KB.

Our method proceeds in iterative rounds as follows. First, a set of fact candidates is computed from text by employing OpenIE [Saha and Mausam, 2018, Saha et al., 2017], and disambiguating entities and normalizing quantities [Hoffart et al., 2011, Roy et al., 2015]. Guided by the KB schema and its entities, we first generate a *targeted query* to extract candidate facts from the text corpus, by using a combination of OpenIE for numeric expressions [Saha and Mausam, 2018, Saha et al., 2017] and rules [Roy et al., 2015] with entity disambiguation and quantity normalization [Hoffart et al., 2011]. After each round, the resulting SPO triples are split into a *high-confidence* and a *low-confidence* group, based on classifiers. The first group is considered mostly correct, while the second group mostly noisy due to many false positives. Subsequent rounds refine both sets, with the goal of expanding the *high-confidence* set.

To further increase the *precision* of the *high-confidence* group, we devise a new technique that captures the value distribution of the quantity of interest and employs a self-consistency test to demote assertions with extremely low likelihood.

To gradually improve the **recall** of the final results, we devise a new technique to expand the targeted query for gathering candidates, after each round. By analyzing contextual cues in the corpus occurrences of the current high-confidence candidates, our method learns informative expansion terms for reformulating the query. For example, given the initial query "⟨entity⟩ height" (with ⟨entity⟩ replaced by the KB entities), we may find snippets where the same relation is expressed by phrases like "stands . . . tall" and can then generate the query "⟨entity⟩ stands tall" for the next round.

By combining the distribution-based denoising with relation contextualization for query expansion, we successfully tackle the goal of high coverage without degenerating the KB quality standard in terms of correctness.

The key contributions of this paper are as follows:

- We present the first approach to scalably capturing quantity facts towards closing gaps in state-of-the-art knowledge bases.
- Our method includes two novel techniques for increasing recall, by automated query expansion, and ensuring high precision, by self-consistency checks against the quantity value distribution.
- Experiments with the Wikidata KB and a text corpus of more than 13 million documents demonstrate the viability of our method. For a set of 6 major predicates, we are able to extract a total of 11,687 quantity facts, with average precision of almost 80%, and above 30% of them missing in the KB. We make the code and data available to the research community[2].

[2]https://www.mpi-inf.mpg.de/research/quantity-search/ql

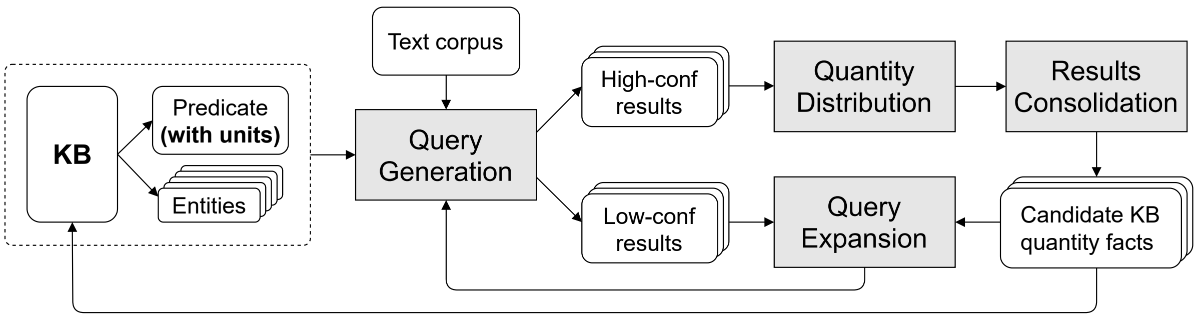


Figure 5.1: QL system overview.

5.2 Overview of the QL System

Given a predicate of interest and a set of KB entities, for which the respective quantities are missing in the KB, our goal is to extract these quantities from a text corpus. The QL system does this in the following steps, as depicted in Figure 5.1.

5.2.1 Step 0: Data Preparation

As pre-processing, we run Open IE [Saha and Mausam, 2018, Saha et al., 2017] on the corpus, recognize and disambiguate named entities (using [Hoffart et al., 2011] for entity linking, and [Lee et al., 2018] for coreference resolution), and detect and normalize quantities (using [Roy et al., 2015]). The output is cast into the **Qfact** representation following [Ho et al., 2019]: tuples of the form $\mathcal{F} = (e, q, X)$ where e is a KB entity, q is a quantity with a numeric value and a mapped-to-KB unit (the latter absent for mere counts), X captures context in the form of a (small) set of cue words that are informative for understanding the relation between e and q.

Example 5.2.1. Given the text snippet *"The Eiffel Tower is 1,063 ft high and costs about $1.5 million to construct."* with disambiguated entity: *"Eiffel Tower"* $\rightarrow$ ⟨*Eiffel_ Tower*⟩ and quantities: *"1,063 ft"* $\rightarrow$ *(1063, feet)* and *"$1.5 million"* $\rightarrow$ *(1500000, $)*; Open IE generates two tuples *(The Eiffel Tower; is; 1,063 ft high)* and *(The Eiffel Tower; costs; about $1.5 million; to construct)*. Mapping them with the entities and quantities and dropping all stop words, we obtain:
- $\mathcal{F}_1$: e = ⟨*Eiffel_ Tower*⟩; q = *(1063, feet)*; X = *"high"*
- $\mathcal{F}_2$: e = ⟨*Eiffel_ Tower*⟩; q = *(1500000, $)*; X = *"costs construct"* □

In contrast to non-numerical data, numerical information is usually mediated by one or a handful of keywords [Madaan et al., 2016]. E.g., sentences about "inflation rate", "GDP", "life expectancy" would be very often accompanied by the respective keywords.

This observation is adopted by the Qfact model, where the context X comprises tokens important for understanding the entity-quantity relation. We use Qfacts as a proxy for populating KB facts.

Our main contribution concerns automatically selecting, out of the set of Qfacts obtained in Step 0, those for which the context indicates the predicate of interest. To this end, we propose a query-driven technique as described next.

5.2.2 Step 1: Predicate-targeted Query Generation

The collected pool of candidate Qfacts are filtered and ranked by a *predicate-targeted query*, generated for the predicate at hand (e.g., *height*) leveraging the KB schema, i.e., the type signatures of predicates and the required units of quantities.

Definition 5.2.1 (Predicate-targeted Query). *The* predicate-targeted query p *is a tuple* $T(p) = (pd, pu, p\mathcal{X})$*, where:*
- pd *is the predicate domain from the KB schema (e.g.,* building*);*
- pu *is a set of possible units for the predicate values (e.g.,* meter, feet*);*
- $p\mathcal{X} = \{pX_0, pX_1, ...\}$ *is the query context – a multiset where each* pX_i *is a bag of words expressing the predicate* p *(e.g., "*height*", "*stands tall*", etc.).*

In the first iteration of QL, we construct the initial targeted query $T_0(p)$ with the fixed domain pd and units pu, taken from the KB schema, and the context comprising of only $p\mathcal{X} = \{pX_0\}$, with pX_0 being the KB label of the predicate (e.g., "height"). Subsequent iterations expand the $p\mathcal{X}$ set with further context tokens to achieve higher recall (e.g., $p\mathcal{X} = \{\text{"height"}, \text{"stands tall"}, \text{"rise"}, ...\}$).

The targeted query is used to rank the candidate Qfacts in terms of their semantic relatedness. More specifically, we compute the relevance score of a Qfact $\mathcal{F} = (e, q, X)$ with respect to the query $T(p) = (pd, pu, p\mathcal{X})$ as:

$$rel(\mathcal{F}, T(p)) = \begin{cases} \max\limits_{pX_i \in p\mathcal{X}} sim(X, pX_i), & \text{if } e \in pd,\ q \in pu \\ 0, & \text{otherwise} \end{cases} \tag{5.1}$$

where sim denotes the semantic similarity between two bags of words. While various options for the choice of sim exist, we use the *context-embedding-distance* of [Ho et al., 2019], which is based on word embeddings. The introduced relevance score ranks all Qfacts, whose entity and quantity match with the domain and units of the target predicate, based on the semantic embedding distance between their context and the best-matched context in the query.

Based on a confidence-threshold parameter γ, we divide the ranked list of Qfacts into a *high-confidence* group H with the score $rel(\mathcal{F}, T(p)) \geq \gamma$ and a *low-confidence* group L with $rel(\mathcal{F}, T(p)) < \gamma$. For setting γ in a principled way, we employ the Deep Open Classification (DOC) method with Gaussian fitting [Shu et al., 2017], using distant supervision from a small set of ground-truth facts of the target predicate extracted from Wikidata. In practice, if no ground-truth facts are available, we could simply set the threshold γ to a high value, to ensure that the *high-confidence* group has mostly accurate results with high probability.

5.2.3 Step 2: Quantity Distribution

The majority of Qfacts in the *high-confidence* group are assumed to be likely correct: capturing the target predicate and having reasonable quantity values. However, a small fraction could still be spurious. To filter these out, we devise a denoising technique (Section 5.3), based on characterizing the value distribution of the *high-confidence* group. The idea is to spot outliers that are likely incorrect, such as buildings with height 1 meter or 5 km, which based on commonsense knowledge cannot be associated with building height. This way, we can eliminate many false positives.

5.2.4 Step 3: Results Consolidation

The previous step will still leave some incorrect or inaccurate Qfact candidates, due to the following: (1) for the same entity, different quantities can be stated at different precision levels (e.g., 302 m, ca. 300 m, more than 300 m); (2) different units can cause deviations after conversion (e.g., 1063 ft $\rightarrow$ 320 m); (3) false statements in the original text; (4) time-variant values or otherwise context-dependent differences in values (e.g., company revenues for a certain year or quarter, or for a certain sales region).

To resolve these kinds of noise and conflicts, we group Qfacts for the same entity-predicate pair by temporal scopes, obtained from the text passage via temporal tagging [Strötgen and Gertz, 2016] or the document timestamp if available (e.g., for news articles). Within each of these groups, we select the most frequent value. The resulting Qfacts are the candidates for addition to the KB.

5.2.5 Step 4: Query Expansion

Since our overarching goal is to boost the recall without degrading precision, we reconsider the *low-confidence* group of Qfact candidates. This group might contain some further relevant statements, but additional sophisticated procedures are required for detecting them. To harvest positive instances from the *low-confidence* group, we propose a statistical method for query expansion, which exploits cleaned *high-confidence* facts from the previous step to automatically extend the predicate contexts $p\mathcal{X}$ with additional relevant phrases (Section 5.4).

Example 5.2.2. If (*Eiffel_ Tower, height, 324m*) is added to the KB in Step 3, and *(Eiffel_ Tower, 324m, "stand tall")* is a Qfact from the *low-confidence* pool, our query expansion mechanism collects the tokens *"stand tall"* as a paraphrasing of the target predicate *height*, and the initial targeted query $T_0(p)$ is expanded by setting $p\mathcal{X}$ to $p\mathcal{X} \cup \{$*"stand tall"*$\}$, which results in $T_1(p)$ with this updated context. □

The above steps are repeated until a stopping criterion is met: the query cannot be expanded further, or we have reached the maximum number of iterations k (we set $k = 10$ in our implementation). We hypothesize that the introduced iterative method of breaking the quantity fact extraction task into smaller sub-problems, corresponding to spotting facts with different context phrases generated automatically would allow us to cautiously retrieve portions of likely correct facts for each computed context. Subsequently combining the results for every sub-problem should yield high overall recall.

5.3 Distribution-based Denoising

This section describes the denoising of the *high-confidence* group of Qfacts. To this end, all quantity values are normalized, by converting to the same standard unit (e.g., meters for height) and combining Qfacts with small differences between their normalized values (within less than 5 percent, e.g., taking the most frequent one from values like 300, 302 and 310 meters for the Eiffel Tower).

Given normalized values from the *high-confidence* group H of Qfacts, the goal is to filter out noisy values from H, based on the value distribution. The key idea is to compute the change in the distribution if a certain value is removed from H. Specifically, for each value $v \in H$, we compute two likelihood scores: (1) the *original likelihood score (o-score)* is the likelihood of v generated from the distribution constructed from the full set of values H (including v); and (2) the *consistency likelihood score (c-score)* is generated from the distributions constructed from random subsets of H excluding v, computed based on a consistency learning technique. We consider the value v as noise if these two scores differ by a substantial amount:

$$\textit{noise-score}(v) = \frac{|\textit{o-score}(v) - \textit{c-score}(v)|}{\max(|\textit{o-score}(v)|, |\textit{c-score}(v)|)}$$

All Qfacts in the *high-confidence* group for which *noise-score*$(v) \geq \mu$ for threshold parameter μ are filtered out.

5.3.1 Original Likelihood Score

For computing the original likelihood score *(o-score)*, we first construct a distribution f from H, with f being the probability density function (PDF), using Kernel Density Estimation (KDE):

$$f(v) = \frac{1}{|H| * b} \sum_{v' \in H} \Phi\left(\frac{v - v'}{b}\right)$$

where Φ is the kernel function. We use a Gaussian kernel, defined as $\Phi(x) = \frac{1}{\sqrt{2\pi}} e^{\frac{-x^2}{2}}$, with bandwidth parameter b. We adopt the state-of-the-art method Improved-Sheather-Jones [Botev et al., 2010] for the automatic choice of the optimal bandwidth.

The *o-score* of value $v \in H$ is then defined as:

$$o\text{-}score(v) = P(f \to v) = \int\limits_{x:f(x) \leq f(v)} f(x)dx \tag{5.2}$$

In other words, we define the likelihood of v as the integral of f over all values whose density is not greater than $f(v)$. As the KDE could have multiple local extrema, we approximate this integral using Simpson's rule with segmentation.

5.3.2 Consistency Likelihood Score

For computing the *consistency likelihood score (c-score)*, we devise a technique inspired by earlier work on consistency learning [Yagnik and Islam, 2007] originally developed for image classification.

Intuitively, this is a form of self-validation, similar to the principle of cross-validation. We randomly sample a small probe set of values from H (10% of H in our implementation), and use the remaining values to construct a distribution. The constructed distribution is then used to measure the likelihood scores of the values in the probe set. This sampling and cross-validation process is repeated a large number of times. The consistency likelihood score (*c-score*) of a value v is computed as the average predicted likelihood, aggregated over all cases where v was in the probe set.

At each sampling iteration, the distribution construction and the value likelihood inference for the *c-score* are similar as for the *o-score*. The only difference is that the optimal bandwidth value b of f constructed from H when computing *o-score* is also used

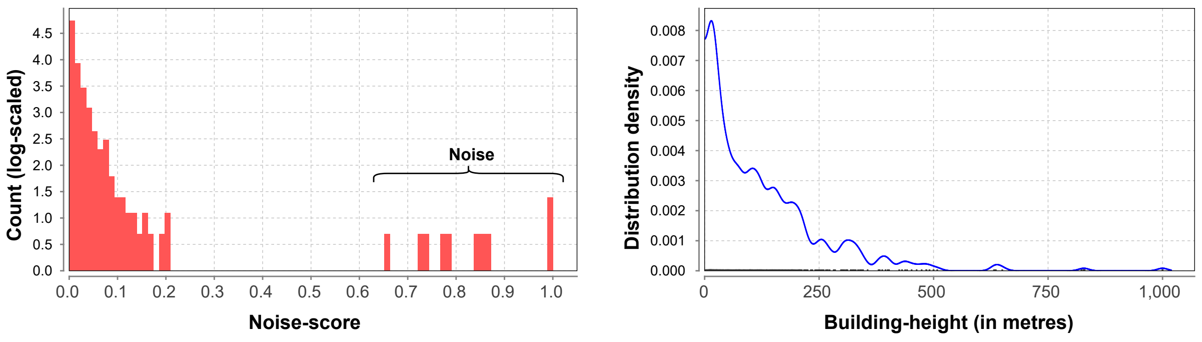


Figure 5.2: *(Left):* Histogram of noise-scores. *(Right):* Reconstructed distribution after denoising.

for constructing distributions from sample subsets of H. We hypothesize that the added noise changes only the shape of the distribution (defined by the samples), but not its smoothness (defined by b).

5.3.3 Denoising Output

The denoising has two results. First, we obtain the *positive* results H^+ after removing all noisy Qfacts from the high-confidence group H, which have a high *noise-score* $\geq \mu$. In experiments, we set μ to 0.3. The positive results H^+ are subsequently consolidated and considered for addition to the KB. Second, we obtain a better estimation of the distribution f from H^+, which is used for query expansion, as described in the next section. For illustration, Figure 5.2 depicts the denoised output of the example predicate *building_height*.

5.4 Query Expansion

At this stage, we have cleaned the *high-confidence* group H of Qfacts and collected the positive results into H^+. While the Qfacts in the *low-confidence* group L are ranked low based on the semantic similarity function (5.1) from Section 5.2, we do not know whether they are actually wrong, thus we treat them as unknown.

In this section, we describe our approach for expanding the predicate-targeted query to achieve a better coverage of the fact extraction process. Specifically, with the current predicate-targeted query at round i: $T_i(p) = (pd, pu, p\mathcal{X} = \{pX_0, ..., pX_i\})$, we learn a candidate context pX', which is then used to expand the query context for the next iteration.

Our query expansion technique relies on the redundancy in the data, i.e., the presence of the same entity and approximately similar quantities in both H^+ and L. To this end, we define the notion of *supported Qfacts*:

Definition 5.4.1 (Supported Qfact). *A given Qfact $\mathcal{F} = (e, q, X)$ from the* low-confidence *group L is supported if in the cleaned high-confidence group H^+ there exists a Qfact $\mathcal{F}' = (e, q', X')$, such that $q \approx q'$ (i.e., $\mathcal{F}'$ has the same entity and approximately the same quantity as $\mathcal{F}$, after conversion to the same standard unit). The* supported set *is the set of all supported facts in L, which we denote by supp-set(L, H^+).*

Example 5.4.1. Consider the *high-confidence* group H^+ with two Qfacts: $H^+ = \{($*Eiffel_ Tower, 324 m, "height"*$)$, $($*Burj_Khalif, 2717 ft, "reached height"*$)\}$. The following Qfacts are supported:
- $\mathcal{F}_1 = ($*Eiffel_ Tower, 324 m, "stand tall"*$)$,
- $\mathcal{F}_2 = ($*Eiffel_ Tower, 1062 ft, "rise"*$)$,
- $\mathcal{F}_3 = ($*Burj_Khalifa, 2722 ft, "originally tall"*$)$ and
- $\mathcal{F}_4 = ($*Burj_Khalifa, 828 m, "rise height"*$)$.

In contrast, the following facts are *not* supported:
- $\mathcal{F}_5 = ($*The_Shard, 1017 ft, "tall"*$)$,
- $\mathcal{F}_6 = ($*Sydney_ Tower, 309 m, "stand high"*$)$ and
- $\mathcal{F}_7 = ($*Eiffel_ Tower, 328 ft, "base wide"*$)$.

The entities of $\mathcal{F}_5$ and $\mathcal{F}_6$ do not appear in H^+, while the quantity of $\mathcal{F}_7$ deviates too much. Given $L = \{\mathcal{F}_1, \dots, \mathcal{F}_7\}$ from above, its supported set is as follows: *supp-set*$(L, H^+) = \{\mathcal{F}_1, \mathcal{F}_2, \mathcal{F}_3, \mathcal{F}_4\}$. □

For each candidate context pX' appearing in the *low-confidence* group L, we can compute the number of statements in L with this context that rephrase facts from the *high-confidence* group H^+. To this end, we define the notion of support as follows:

Definition 5.4.2 (Support). *Given candidate context pX', its support is the number of Qfacts in the supported set of L whose context includes pX'.*

$$\textit{supp}(pX', L, H^+) = |\{(e, q, X) \in \textit{supp-set}(L, H^+) : pX' \subseteq X\}|$$

Example 5.4.2. For the *high-confidence* group H^+ and the *low-confidence* group L of Example 5.4.1, we have: *supp*$($*"stand"*$, L, H^+) = |\{\mathcal{F}_1\}| = 1$, *supp*$($*"tall"*$, L, H^+) = |\{\mathcal{F}_1, \mathcal{F}_3\}| = 2$, and *supp*$($*"rise"*$, L, H^+) = |\{\mathcal{F}_2, \mathcal{F}_4\}| = 2$. In general, the candidate context pX' is not limited to single tokens. For example, it holds that *supp*$($*"rise height"*$, L, H^+) = |\{\mathcal{F}_4\}| = 1$. □

We remove all candidate contexts with support lower than a pre-defined threshold. High support is important, but it is not sufficient for a candidate context to be a paraphrase or refinement of the original predicate p. Indeed, many uninformative words also have high support, for example "about", "during", "up", etc. These are words with low *inverse document frequency (idf)*, which can be filtered out by thresholding.

Support can be normalized by the highest support value among all the candidate contexts. We define the *relative support* of a candidate context as:

$$\textit{r-supp}(pX', L, H^+) = \frac{\textit{supp}(pX', L, H^+)}{\max_{pX''} \textit{supp}(pX'', L, H^+)}$$

To effectively select promising candidate contexts for query expansion, we additionally take the quantities of the respective statements into account by exploiting the following proposed measures.

Definition 5.4.3 (Expansion Set). *Given candidate context pX' and the* low-confidence *group L, the expansion set of pX' includes all Qfacts in L whose context contains pX':*

$$\textit{exp-set}(pX', L) = \{(e, q, X) \in L \,|\, pX' \subseteq X\}$$

Example 5.4.3. Consider the *low-confidence* group L as in Example 5.4.1. We have: $\textit{exp-set}(\text{"stand"}, L) = \{\mathcal{F}_1, \mathcal{F}_6\}$, $\textit{exp-set}(\text{"tall"}, L) = \{\mathcal{F}_1, \mathcal{F}_3, \mathcal{F}_5\}$ and $\textit{exp-set}(\text{"rise"}, L) = \{\mathcal{F}_2, \mathcal{F}_4\}$. $\square$

Intuitively, the expansion set comprises all Qfacts in the *low-confidence* group that contain pX', regardless of whether they are supported by any of the facts in the *high-confidence* group or not. These are potential statements that could be added to the *high-confidence* group if pX' is chosen to expand the query.

To measure the quality of an expansion set, we compare the quantity values of its Qfacts to the value distribution f as estimated in Section 5.3.

Definition 5.4.4 (Distribution Confidence). *The distribution confidence is the average likelihood of the quantity values in the expansion set, generated by the distribution f constructed from the* high-confidence *group H^+:*

$$\textit{d-conf}(pX', L, H^+) = \frac{1}{|\textit{exp-set}(pX', L)|} \sum_{(e,q,X) \in \textit{exp-set}(pX', L)} P(f \to q)$$

where $P(f \to q)$ is the integral function of Equation 5.2.

Intuitively, a good candidate context for paraphrasing or refining the original predicate should have an expansion set whose quantities comply with the reference distribution.

As a second signal for scoring the suitability of an expansion set, we use the original relevance scores of its Qfacts:

Definition 5.4.5 (Querying Confidence). *The querying confidence is the average relevance score of the Qfacts in the expansion set relative to the predicate-targeted query $T_i(p)$ at the given round:*

$$\textit{q-conf}(pX', L) = \frac{1}{|\textit{exp-set}(pX', L)|} \sum_{\mathcal{F} \in \textit{exp-set}(pX', L)} \textit{rel}(\mathcal{F} | T_i(p))$$

where rel is the function from Equation 5.1.

Finally, for a candidate context pX', its suitability for expanding the original query is computed using the following score:

Definition 5.4.6 (Candidate Expansion Score). *The expansion score of a candidate context pX' is a (hyperparameter-)weighted sum of the relative support, the querying confidence, and the distribution confidence:*

$$\begin{aligned}\textit{expansion-score}(pX', L, H^+) = w_1 \cdot \textit{r-supp}(pX', L, H^+) + \\ w_2 \cdot \textit{d-conf}(pX', L, H^+) + w_3 \cdot \textit{q-conf}(pX', L)\end{aligned}$$

Based on the *expansion-score*(pX', L, H^+) value, we rank candidate contexts pX' appearing in the *low-confidence* group L, and select the best one to expand the query for the next iteration.

5.5 Experiments

We evaluated the quality of the QL output in two settings: *KB augmentation*, our major use case, and *quantity search* over web pages, as an extrinsic use case.

5.5.1 Experimental Setup

Input Data. We focus on the Wikidata KB, as it is the richest, among large publicly available KBs, in terms of quantity facts. We selected the following six numerical predicates, covering a spectrum of topical themes, with big gaps between potential facts and Wikidata coverage : *building-height (P2048)*, *mountain-elevation (P2044)*, *stadium-capacity (P1082)*, *river-length (P2043)*, *powerstation-capacity (P2109)*, and *earthquake-magnitude (P2528)*. We take entities from their corresponding domain types: *building*, *mountain*, *stadium*, *river*, *powerstation* and *earthquake*, respectively. These also include entities for which Wikidata has no triples on the above target predicates. We restrict ourselves to entities which are linked to a Wikipedia page, as only these are supported by the linking system [Hoffart et al., 2011] that we rely on. Our goal is to acquire quantity values for these entities and predicates, regardless of whether Wikidata already has the target values or not. As input text corpus, we obtained the collection used in [Ho et al., 2019, Ho et al., 2020] from the authors. This textual corpus comprises ca. 13 million web pages from the English Wikipedia and from news articles.

Quality Metrics. We measure the quality of the extracted quantity facts using the following three metrics: *precision*, *recall*, and *novelty*. For computing precision and recall, we use the Wikidata KB as ground-truth. Let G_p denote the set of triples stored in Wikidata for the target predicate p, and let S_p denote the set of facts extracted using our QL method. Then, precision is computed as:

$$prec(S_p) = \frac{|S_p \cap G_p| + |S_p \backslash G_p| \times prec_{sampled}(S_p \backslash G_p)}{|S_p|}$$

where $S_p \cap G_p$ and $S_p \backslash G_p$ are the sets of facts extracted by our method, that are inside and outside the ground-truth, respectively. We estimate the correctness of $S_p \backslash G_p$ by randomly sampling 30 facts and computing their precision (i.e., $prec_{sampled}$).

Recall is computed relatively to what Wikidata already contains as follows:

$$recall(S_p) = \frac{|S_p \cap G_p|}{|G_p|}$$

Obviously, this measure misses a key point that our QL system can acquire new facts from web sources that are absent in Wikidata. To evaluate this dimension, we compute the *novelty* metric as the fraction of correctly extracted facts, which are outside of the groundtruth set. This measure demonstrates the effectiveness of our approach for extending existing KBs with new knowledge:

$$novelty(S_p) = \frac{|S_p \backslash G_p| \times prec_{sampled}(S_p \backslash G_p)}{|S_p|}$$

Baselines. We compare the QL method to the following three baselines: our prior work *Qsearch* [Ho et al., 2019] and two methods based on pre-trained language models (*LMs*), RoBERTa [Liu et al., 2019b] and GPT-3 [Brown et al., 2020].

- *Qsearch.*[3] Qsearch is designed to answer quantity-filter queries such as "buildings higher than 1000 ft". For each predicate, we manually create an input query for Qsearch, setting the filter condition to "$>$ 0", to obtain the highest yield. Qsearch returns a list of confidence-ranked answers. For fair comparison, we consider only the top-N answers of Qsearch, where N is the number of facts extracted by QL.

- *LM RoBERTa.*[4] We use mask prediction for each target entity, exploiting per-predicate templates. For example, for building-height the template is: *"[ENTITY] has a height of [VALUE] [UNIT]"*, where "[ENTITY]" is the surface form of the entities and "[UNIT]" is filled from the units pu of the predicate-targeted query, like meter and feet for building-height. Due to the limitation of the mask prediction task, in this mode, the LM can predict only a single token, hence the need for helping by giving the unit as input. We let RoBERTa predict the value for each of the possible units, and treat the output as correct if the predicted value is correct for one of the units. In addition, we consider a *RoBERTa@5* configuration, where the output is considered correct if the correct value is among not just the top-1 but top-5 predictions.

- *GPT-3.*[5] GPT-3 has versatile interfaces including question answering. We probe GPT-3 with a short input text, beginning with 3 examples of question/answer pairs for target predicate and ending with an explicit question that GPT-3 has to answer. This is based on a hand-crafted template for each predicate. An example input is:

  How tall is the Eiffel Tower? 324 meters.

  How tall is Burj Khalifa? 2,717 feet.

  How tall is Empire State Building? 381 m.

  How tall is [ENTITY]?

  where "[ENTITY]" is replaced by each of the target entities. Since GPT-3 is able to generate multi-token answers (which is in contrast to RoBERTa), we do not encode the target units in the input. The quantity extractor [Roy et al., 2015] is then used to parse the GPT-3 output to obtain the quantity result.

[3] https://qsearch.mpi-inf.mpg.de/
[4] https://huggingface.co/roberta-large
[5] https://beta.openai.com/

Predicate	*Method*	*#Facts*	*Prec.(%)*	*Rec.(%)*	*Nov.(%)*
building-height	Qsearch	1253	56.14	9.91	48.48
	RoBERTa	1253	7.02	2.06	5.42
	RoBERTa@5	1253	20.35	5.99	15.72
	GPT-3	1253	4.56	1.34	3.52
	QL	1253	**82.46**	**24.25**	**63.70**
mountain-elevation	Qsearch	3244	57.34	20.32	10.64
	RoBERTa	3289	1.76	0.64	0.30
	RoBERTa@5	3289	8.16	2.98	1.41
	GPT-3	3289	5.99	2.19	1.03
	QL	3289	**91.36**	**33.35**	**15.78**
stadium-capacity	Qsearch	3496	31.10	19.25	16.51
	RoBERTa	3496	1.18	0.91	0.49
	RoBERTa@5	3496	6.58	5.06	2.75
	GPT-3	3496	2.99	2.30	1.25
	QL	3496	**70.10**	**53.91**	**29.26**
river-length	Qsearch	3019	14.36	5.82	5.84
	RoBERTa	3019	4.14	1.27	2.28
	RoBERTa@5	3019	19.05	5.84	10.51
	GPT-3	3019	1.11	0.34	0.61
	QL	3019	**60.71**	**18.62**	**33.48**
powerstation-capacity	Qsearch	319	53.50	7.12	27.17
	RoBERTa	394	2.75	0.51	1.23
	RoBERTa@5	394	15.14	2.80	6.76
	GPT-3	394	5.96	1.10	2.66
	QL	394	**75.23**	**13.90**	**33.60**
earthquake-magnitude	Qsearch	236	53.33	16.49	46.55
	RoBERTa	236	23.08	12.37	17.99
	RoBERTa@5	236	67.31	36.08	52.48
	GPT-3	236	38.46	20.62	29.99
	QL	236	**88.46**	**47.42**	**68.97**

Table 5.1: #Facts, precision, recall, novelty for our proposed method QL and baselines.

5.5.2 Results for KB Augmentation

Table 5.1 shows the main results: precision, recall and novelty for each predicate, as achieved by the methods under comparison. It also reports the total number of extracted quantity facts per predicate.

The total number of extracted facts by our QL method varies from hundreds to thousands, with precision reaching ca. 90 percent for the best cases. This shows the great potential for augmenting a high-quality KB with additional quantity facts. For some predicates, the precision is considerably lower, pointing out the need for further research. Nonetheless, we could choose more conservative thresholds to boost precision for a subset, at the expense of losing some recall. In any case, the QL method outperforms all baselines on precision by a huge margin (with Qsearch being second-best).

The recall, relative to Wikidata facts, is decent, but exhibits that QL could not find

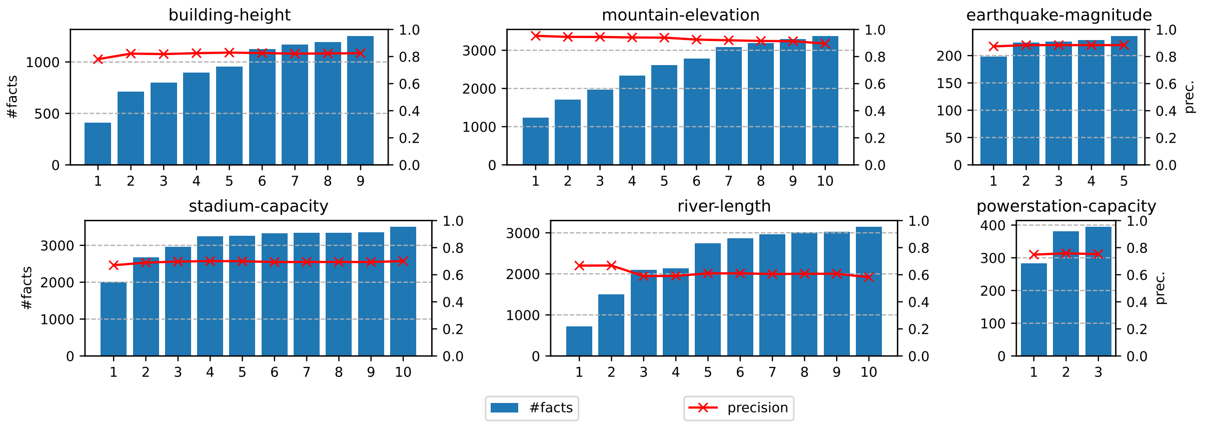


Figure 5.3: #Facts and precision after each round.

Predicate	Expanded query context in each iteration
building-height	1) *"height"*, 2) *"tall"*, 3) *"tallest"*, 4) *"rise"*, 5) *"skyscraper"*, 6) *"high"*, 7) *"build"*, 8) *"build tallest"*, 9) *"stand"*
mountain-elevation	1) *"elevation"*, 2) *"peak"*, 3) *"highest"*, 4) *"high"*, 5) *"level sea"*, 6) *"rise"*, 7) *"height"*, 8) *"summit"*, 9) *"altitude"*, 10) *"locate"*
stadium-capacity	1) *"capacity"*, 2) *"hold"*, 3) *"seat"*, 4) *"spectator"*, 5) *"people"*, 6) *"stadium"*, 7) *"seat stadium"*, 8) *"hold people"*, 9) *"capacity seat"*, 10) *"multi-purpose seat"*
river-length	1) *"length"*, 2) *"tributary"*, 3) *"flow"*, 4) *"state"*, 5) *"river"*, 6) *"stream"*, 7) *"run"*, 8) *"long tributary"*, 9) *"longest"*, 10) *"north"*
powerstation-capacity	1) *"capacity"*, 2) *"power"*, 3) *"generate"*
earthquake-magnitude	1) *"magnitude"*, 2) *"measure"*, 3) *"scale"*, 4) *"moment"*, 5) *"estimate"*

Table 5.2: Expansion of predicate-targeted queries per round.

many of the Wikidata entities in its text corpus. This could be a limitation of what the corpus covers, and how difficult extraction from text passages is. The latter point holds particularly for Wikipedia articles, which often have complex sentences and long paragraphs with many pronouns rather than explicit entity mentions.

Finally, the novelty numbers underline the great opportunity to add new quantity facts to the KB: enriching entities with quantities that are so far absent. This potential is most pronounced for *building-height* and *earthquake-magnitude* predicates. Wikidata seems to contain many long-tail entities of these types, but misses out on the crucial quantity facts. Again, QL excels against all baselines in this regard.

To investigate whether the iterative approach of QL is productive, we also report the extraction quality measures after each round. Figure 5.3 plots the number of extracted facts and the precision per iteration. We do indeed see a steadily increase in the output size, thus acquiring more facts after each round. The precision lines stay fairly high throughout these iterations, showing that we barely lose precision while advancing recall and novelty. Table 5.2 reports the query context automatically expanded in each iteration of our method.

Run-Time. With our implementation, the total fact extraction time for each predicate ranges from one to fifteen minutes, depending on the number of executed iterations (from three to ten), and facts extracted from them (from 236 to almost 3500, for predicates *powerstation-capacity* and *stadium-capacity*, respectively).

Note that our method runs offline in batch mode. For scalability, the method can be easily parallelized by data partitioning.

5.5.3 Effectiveness of Quantity De-noising

Evaluation Setup. In this subsection, we evaluate the effectiveness of our distribution-based de-noising procedure. For that, we compare the quality of Qfacts from the high-confidence group pruned using our distribution-based denoising procedure, to the quality of Qfacts pruned using competitive methods. The quality is defined as the percentage of pruned Qfacts that are actually noisy (i.e., the higher, the better).

Baselines. Our baselines include *Range@1*, *Percentile@1*, *DB-scan*, *Z-score* and *DoQ*. For all baselines, the quantities are converted to the same unit before de-noising.

- *Range@1.* This baseline divides the range of all Qfacts in the *high-confidence* group H into 100 equal intervals, and considers values belonging to the first or the last intervals as noise. More specifically, the value v is treated as noise if $v \notin (min(H) + \frac{1}{100}range(H), max(H) - \frac{1}{100}range(H))$, where $range(H) = max(H) - min(H)$.

- *Percentile@1.* The *Percentile@1* baseline always considers top 1% largest and 1% smallest values of the ranked list of Qfacts in the *high-confidence* group S as noise.

- *Z-score.* The *Z-score* baseline treats an extraction as noise if its z-score computed from the empirical distribution is more than 3 or less than -3, which are standard values for detecting outliers.

- *DB-scan.* The *DB-scan* baseline first normalizes the numeric value set by dividing every value in the set by the highest absolute value, and then running the DB-scan

Predicate	*Rng@1*	*Prct@1*	*DBscan*	*Z-scr*	*DoQ*	*QL*
building-height	1334 (25.1)	28 (100.0)	1 (100.0)	5 (100.0)	176 (49.8)	28 (96.4)
mountain-elevation	2800 (9.9)	64 (81.8)	3 (100.0)	6 (100.0)	8 (100.0)	34 (91.2)
stadium-capacity	2881 (25.2)	76 (78.9)	0 –	6 (83.3)	× –	150 (98.4)
river-length	2637 (29.2)	60 (75.0)	1 (100.0)	26 (42.3)	925 (30.5)	71 (66.2)
powerstation-capacity	410 (21.1)	8 (50.0)	1 (100.0)	1 (100.0)	60 (38.3)	0 –
earthquake-magnitude	251 (23.5)	5 (100.0)	1 (100.0)	2 (100.0)	× –	10 (100.0)

(×) Not applicable, since the distribution is not provided by DoQ.

Table 5.3: Denoising results. *#predicted_noise (%actual_noise)*

algorithm to detect outliers (with default parameters $eps = 0.5$ and $min\_samples = 5$, provided by the scikit-learn package[6]).

- *DoQ*. Finally, the *DoQ* baseline is similar to *Z-score*, except that the reference distribution is provided from the DoQ dataset [Elazar et al., 2019].

Evaluation Results. Table 5.5.3 reports the results for this experiment. More specifically, we present the number of Qfacts whose values have been classified as noise by every method with the quality of such classification provided in brackets – the percentage of pruned Qfacts that are actually noisy, hence the more facts and the higher quality, the better. Note that these reported numbers are of the Qfacts, not KB facts – the results before the consolidation step 3. We observe that our de-noising method outperforms all baselines for all predicates except river-length, for which we predict more noise than the baseline Prct@1, but with slightly lower precision. To investigate the reasons behind such results, we have additionally performed the Anderson-Darling (AD) test for each predicate by measuring the p-value of the values in the *high-confidence* group regarding the KDE distribution constructed from it. The AD-test is used to measure how likely the considered values actually come from the target distribution. The results are shown in Table 5.4. From this table, we can confidently reject the null hypothesis for the predicates *river-length* and *powerstation-capacity*, in which we cannot construct an

[6]https://scikit-learn.org

Predicate	*p-value*	Predicate	*p-value*
building-height	0.124	river-length	≈ 0.00
mountain-elevation	0.096	powerstation-capacity	≈ 0.00
stadium-capacity	0.139	earthquake-magnitude	0.787

Table 5.4: Anderson-Darling test results.

accurate KDE distribution from the given values. This explains the results achieved in Table 5.5.3.

5.5.4 Extrinsic Use Case: Quantity Search

We aim to compute top-ranked answers with quantity filter conditions such as: "buildings with height above 1000 ft", or "sprinters who ran 100 meters under 9.9 seconds". To this end, we first run the QL extraction pipeline, and then evaluate the queries against its output.

Baselines. The original Qsearch system also serves as our main baseline. In addition, we compare to results obtained from top-10 result-page snippets from Google. These are evaluated manually, by considering a snippet as correct if it contains a correct entity and a quantity that satisfies the query filter. This gives Google an advantage, as it does not have to explicitly extract the target entity and quantity.

Benchmark. We use the benchmark queries Q150 from the work [Ho et al., 2021a]. This comprises 150 queries spanning the following four domains: finance, transport, sports, and technology. Each query has ground-truth answer entities, along with their quantity values, based on manually identified Wikipedia list pages.

Results. Table 5.5 reports the results for this experiment: *precision@10*, *recall@10* and *mean average precision (mAP)* over the top-10 ranks. The table shows that QL can indeed improve on the – already very good – results of Qsearch, on all metrics. Both QL and Qsearch outperform the search-engine baseline, which underlines the need for this research.

5.6 Related Work

Numerical Fact Detection. Detecting numerical expressions with units in textual data has been well addressed in prior works [Sarawagi and Chakrabarti, 2014, Alonso

System	*Prec.@10*	*Recall@10*	*mAP@10*
Google	0.167	0.076	0.041
Qsearch	0.290	0.177	0.119
QL	0.301	0.189	0.129

Table 5.5: Quantity search results.

and Sellam, 2018, Roy et al., 2015]. This alone is insufficient, though: for quantity facts, we also need to infer to which entity the expression refers.

The NumberTron method [Madaan et al., 2016] specifically tackled numerical facts for geo-political entities, using a probabilistic graphical model. However, the approach does not scale to large text corpora and achieves only moderate precision.

The authors of [Sarawagi and Chakrabarti, 2014] proposed the QEWT method for answering numerical lookup queries such as *"co2 emissions of china"*, or *"net worth of zuckerberg"*. This work uses only data from web tables and does not carry over to text input. Moreover, it does not directly solve the problem of KB population as we do.

To this end, the work on Qsearch [Ho et al., 2019, Ho et al., 2020] customized a distantly supervised LSTM network for extracting quantities, entities and their context from individual sentences. However, Qsearch is geared for capturing a small number of top-ranked facts as responses to quantity-filter queries. Beyond the top ranks, its precision degrades drastically, by design. In contrast, our method balances larger recall with high precision, for the goal of augmenting a high-quality KB.

Recent works on numerical IE include [Cai et al., 2019, Cao et al., 2021, Chen et al., , Elazar et al., 2019, Mehta et al., 2021]. They tackle a variety of specific settings such as commonsense assertions (e.g., "lions have 4 legs"), properties of commercial products (e.g., price, shipping weight), or quantities in clinical narratives and patient records (e.g., lab values or drug dosages). All of these prior works focus on the IE task itself, without consideration of a KB.

Numerical Embeddings. In [Jiang et al., 2020, Spithourakis and Riedel, 2018, Wallace et al., 2019, Sundararaman et al., 2020, Naik et al., 2019] word embeddings are adapted to account for numerical expressions. The work [Cao et al., 2018] uses bi-directional and DAG-structured LSTM networks to learn simple formulas from verbal descriptions of numerical claims. None of these methods address the task of extracting full-fledge quantity facts, as needed for augmenting a KB.

Language Models for Numeracy. A direction that has gained great attention is the potential role of pre-trained language models as knowledge bases [Petroni et al., 2019, Lin et al., 2020, Heinzerling and Inui, 2021]. The idea is to prompt huge LMs like BERT, GPT-3 or T5 for numerical facts via cloze questions with masked parts to be completed through LM inference [Lin et al., 2020, Berg-Kirkpatrick and Spokoyny, 2020]. However, none of these methods reaches sufficiently high precision to be considered for enhancing a high-quality KB.

Knowledge Graph Completion. Graph-structured KB embeddings [Wang et al., 2017] have been considered for completing gaps in KBs. However, their precision is far from reaching the quality standards of KBs like Wikidata. Moreover, even the exceptional works that specifically tackle numerical predictions [Kotnis and García-Durán, 2019] only consider the knowledge base itself as input. They are not geared for extraction of quantity facts from text.

Iterative learning has been used for fact extraction from text in various works (e.g., [Agichtein and Gravano, 2000, Brin, 1998, Etzioni et al., 2004, Carlson et al., 2010, Suchanek et al., 2009]). However, none of these works is specifically designed for extracting quantity facts.

5.7 Summary

In this chapter, we have presented a method for augmenting knowledge bases with quantity facts extracted from text at large scale. Our method relies on self-consistency checks against the quantity value distribution with the goal of ensuring high precision, and automated query expansion targeted at increasing the coverage of extracted facts. The evaluation of the method's effectiveness showed that it can indeed boost the recall of previous works while retaining high precision, even after several iterative rounds and acquiring thousands of new facts.

Chapter 6

Conclusions

6.1 Summary

Entities with quantities is an important research theme for building next-generation information retrieval and question answering systems. Searching for entities with quantity conditions over web content poses many challenges, including recognizing quantities, extracting quantity facts, reasoning over quantities, denoising, deduplicating, aggregating, and more. This dissertation has developed methods for understanding and reasoning over quantities on the web, for the purpose of answering quantity queries.

The first contribution, Qsearch, is a search and QA system, that can effectively answer advanced queries with quantity conditions from text sources. Our solution is based on a deep neural network for extracting quantity-centric tuples from text sources, and a novel matching model to retrieve and rank answers from news articles and other web pages. Experiments on benchmark queries collected by crowdsourcing demonstrate the effectiveness of our method. A demonstration of Qsearch can be found at https://qsearch.mpi-inf.mpg.de/

The second contribution, QuTE, is 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, leveraging 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. A demonstration of QuTE can be found at https://qsearch.mpi-inf.mpg.de/table/.

The third contribution, QL, is a recall-oriented approach for enhancing knowledge bases with quantity facts. Our method is based on iterative learning for extracting quantity facts, with two novel techniques to boost recall for KB augmentation without sacrificing the quality standards of the knowledge base. The first one is a query expansion technique to capture a larger pool of fact candidates. The second one is a novel technique for harnessing observations on value distributions for self-consistency. Experiments with extractions from more than 13 million web documents demonstrate the benefits of our method.

Experimental data and code of our methods are made publicly available to the research community at the following link: https://www.mpi-inf.mpg.de/departments/databases-and-information-systems/research/yago-naga/quantity-search.

6.2 Outlook

Several prominent future directions could be considered to further advance the research on answering quantity queries. In what follows, we discuss the most important ones.

6.2.1 Answering Complex Quantity Queries

The search systems in this dissertation, Qsearch and QuTE, only handle quantity queries with a single quantity filter condition. However, advanced knowledge workers usually have the demand for searching with more complex queries. For example:

- **Queries with multiple filters:** e.g., *"hybrid cars with a battery range above 50 miles and cost under $100K"*, *"buildings higher than 100m built before 1980"*.

- **Queries with implicit filters:** e.g., *"materials with viscosity lower than that of water"*, *"countries with higher population than Germany"*.

- **Chained queries:** e.g., *"directors of movies with revenue above 1B USD"*.

- **Counting/Superlative queries:** e.g., *"how many sprinters have completed 100-metre race under 10s at least five times in Olympic games?"*, *"which is the most energy-efficient car model?"*.

Answering these kinds of complex queries present further obstacles beyond understanding the quantities. The most obvious challenge is to correctly parse and understand the query semantics, in order to identify the correct resolution logic for the query. For instance, chained queries and queries with multiple and/or implicit filters can be transformed into SPARQL queries, which can be run over knowledge graphs (if they have sufficient coverage). Directly matching queries against web text content requires the capabilities of denoising, reasoning and joining of quantity facts extracted from multiple heterogeneous sources. Finally, handling counting queries and superlative queries raises concerns about the freshness and completeness of the underlying data source, mechanisms for approximating the answers are indispensable, e.g., computing a upper or a lower bound of a counting query.

6.2.2 Quantity Search with Non-KB Results

So far, the works presented in this dissertation only aim at KB entities for extraction and search. However, real-world entities may appear out of the KB scope, especially newly-emerging entities such as products with different models and versions. Searching for out-of-KB entities is an important research direction, with applications for analysts, businesses and e-commerce.

Searching for non-KB results poses additional challenges in processing the entities in the extracted facts, namely the need for understanding the entity type to match against the query type, and detecting duplicates in the extractions to avoid over-counted answers.

6.2.3 Extracting Higher-arity Quantity Facts

Quantity facts in KBs are not restricted to mere triples. Many of these facts should be accompanied by additional qualifiers for clearer semantics, e.g., company-revenue in a specific year, car-fuel-consumption in city/on highway, product-price in a certain country, etc. Even quantity values of simple predicates such as building-height or stadium-capacity may change overtime. Populating KBs with full-qualified quantity facts is beneficial for not only answering complex quantity search queries, but also other downstream applications that leverage KBs.

6.2.4 Extracting Quantity Facts for Special Domains

Quantity facts are necessary not only for encyclopedia KBs, but also for special domains such as biomedical and other scientific KBs, with many important quantitative properties such as drug daily dosage, material coercivity or thermal conductivity.

The challenges of extracting quantity facts from these special domains arise in all steps: named entity recognition and disambiguation (e.g, entities such as *"Ni-*$Ce_{0.9}Gd_{0.1}O_{1.95}$*"*

or *"$[Ni(N_2H_4)_2]Cl_2$"* in material science), quantity recognition (e.g., quantities with complex units such as *"mW/cm^{-2}"* or *"$kJ\ mol^{-1}$"*), coreference resolution, relation extraction, and more. Extracting additional qualifiers for these facts is even more difficult, e.g., liquid-viscosity or material-density at a certain temperature or pressure. Working on specific domains, a fair amount of training and background data is usually needed to create the extraction models. Building such prior knowledge necessitates the involvement of domain specialists.

6.2.5 Other Research Directions

Other research directions revolving around quantities might include quantity fact checking, quantity inconsistency detection as well as KB applications such as quantitative question answering, quantitative rule learning, analysis and reasoning over quantities. Finally, developing methods for dealing with quantities from multilingual inputs is also a relevant research direction.

List of Figures

List of Tables

Bibliography

[Abujabal et al., 2017] Abujabal, A., Roy, R. S., Yahya, M., and Weikum, G. (2017). QUINT: interpretable question answering over knowledge bases. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017 - System Demonstrations*.

[Abujabal et al., 2018] Abujabal, A., Roy, R. S., Yahya, M., and Weikum, G. (2018). Never-ending learning for open-domain question answering over knowledge bases. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*.

[Agichtein and Gravano, 2000] Agichtein, E. and Gravano, L. (2000). *Snowball*: extracting relations from large plain-text collections. In *Proceedings of the Fifth ACM Conference on Digital Libraries, June 2-7, 2000, San Antonio, TX, USA*.

[Akbik et al., 2018] Akbik, A., Blythe, D., and Vollgraf, R. (2018). Contextual string embeddings for sequence labeling. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018*. Association for Computational Linguistics.

[Akbik and Löser, 2012] Akbik, A. and Löser, A. (2012). Kraken: N-ary facts in open information extraction. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction, AKBC-WEKEX@NAACL-HLT 2012, Montrèal, Canada, June 7-8, 2012*. Association for Computational Linguistics.

[Albrecht et al., 2019] Albrecht, J., Belger, A., Blum, R., and Zimmermann, R. (2019). Business analytics on knowledge graphs for market trend analysis. In *Proceedings of the Conference on "Lernen, Wissen, Daten, Analysen", Berlin, Germany, September 30 - October 2, 2019*. CEUR-WS.org.

[Alonso and Sellam, 2018] Alonso, O. and Sellam, T. (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.

[Angeli et al., 2015] Angeli, G., Premkumar, M. J. J., and Manning, C. D. (2015). Leveraging linguistic structure for open domain information extraction. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers.*

[Aralikatte et al., 2019] Aralikatte, R., Lent, H. C., González-Garduño, A. V., Hershcovich, D., Qiu, C., Sandholm, A., Ringaard, M., and Søgaard, A. (2019). Rewarding coreference resolvers for being consistent with world knowledge. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019.* Association for Computational Linguistics.

[Bader et al., 2020] Bader, S. R., Grangel-González, I., Nanjappa, P., Vidal, M., and Maleshkova, M. (2020). A knowledge graph for industry 4.0. In *The Semantic Web - 17th International Conference, ESWC 2020, Heraklion, Crete, Greece, May 31-June 4, 2020, Proceedings.* Springer.

[Baevski et al., 2019] Baevski, A., Edunov, S., Liu, Y., Zettlemoyer, L., and Auli, M. (2019). Cloze-driven pretraining of self-attention networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019.* Association for Computational Linguistics.

[Balog, 2018] Balog, K. (2018). *Entity-Oriented Search.*

[Banerjee et al., 2009] Banerjee, S., Chakrabarti, S., and Ramakrishnan, G. (2009). Learning to rank for quantity consensus queries. In *Proceedings of the 32nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2009, Boston, MA, USA, July 19-23, 2009.*

[Banko et al., 2007] Banko, M., Cafarella, M. J., Soderland, S., Broadhead, M., and Etzioni, O. (2007). Open information extraction from the web. In *IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007.*

[Bast and Haussmann, 2015] Bast, H. and Haussmann, E. (2015). More accurate question answering on freebase. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015.*

[Bender et al., 2003] Bender, O., Och, F. J., and Ney, H. (2003). Maximum entropy models for named entity recognition. In *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003.* ACL.

[Berg-Kirkpatrick and Spokoyny, 2020] Berg-Kirkpatrick, T. and Spokoyny, D. (2020). An empirical investigation of contextualized number prediction. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020.*

[Bhagavatula et al., 2015] Bhagavatula, C. S., Noraset, T., and Downey, D. (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.*

[Bhutani et al., 2016] Bhutani, N., Jagadish, H. V., and Radev, D. R. (2016). Nested propositions in open information extraction. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing, EMNLP 2016, Austin, Texas, USA, November 1-4, 2016.* The Association for Computational Linguistics.

[Bikel et al., 1999] Bikel, D. M., Schwartz, R. M., and Weischedel, R. M. (1999). An algorithm that learns what's in a name. *Mach. Learn.*

[Bollacker et al., 2008] Bollacker, K. D., Evans, C., Paritosh, P. K., Sturge, T., and Taylor, J. (2008). Freebase: a collaboratively created graph database for structuring human knowledge. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2008, Vancouver, BC, Canada, June 10-12, 2008.*

[Botev et al., 2010] Botev, Z. I., Grotowski, J. F., and Kroese, D. P. (2010). Kernel density estimation via diffusion. *The Annals of Statistics.*

[Braunschweig et al., 2015] Braunschweig, K., Thiele, M., and Lehner, W. (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.*

[Brin, 1998] Brin, S. (1998). Extracting patterns and relations from the world wide web. In *The World Wide Web and Databases, International Workshop WebDB'98, Valencia, Spain, March 27-28, 1998, Selected Papers.*

[Brown et al., 2020] Brown, T. B., Mann, B., Ryder, N., Subbiah, M., Kaplan, J., Dhariwal, P., Neelakantan, A., Shyam, P., Sastry, G., Askell, A., Agarwal, S., Herbert-Voss, A., Krueger, G., Henighan, T., Child, R., Ramesh, A., Ziegler, D. M., Wu, J., Winter, C., Hesse, C., Chen, M., Sigler, E., Litwin, M., Gray, S., Chess, B., Clark, J., Berner, C., McCandlish, S., Radford, A., Sutskever, I., and Amodei, D. (2020). Language models are few-shot learners. In *Advances in Neural Information Processing Systems 33: Annual Conference on Neural Information Processing Systems 2020, NeurIPS 2020, December 6-12, 2020, virtual.*

[Cafarella et al., 2018] Cafarella, M. J., Halevy, A. Y., Lee, H., Madhavan, J., Yu, C., Wang, D. Z., and Wu, E. (2018). Ten years of webtables. *Proc. VLDB Endow.*

[Cai et al., 2019] Cai, T. A., Zhang, L., Yang, N., Kumamaru, K. K., Rybicki, F. J., Cai, T., and Liao, K. P. (2019). Extraction of EMR numerical data: an efficient and generalizable tool to EXTEND clinical research. *BMC Medical Informatics Decis. Mak.*

[Cao et al., 2021] Cao, Y., Chen, D., Xu, Z., Li, H., and Luo, P. (2021). Nested relation extraction with iterative neural network. *Frontiers Comput. Sci.*

[Cao et al., 2018] Cao, Y., Li, H., Luo, P., and Yao, J. (2018). Towards automatic numerical cross-checking: Extracting formulas from text. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018.*

[Carlson et al., 2010] Carlson, A., Betteridge, J., Kisiel, B., Settles, B., Jr., E. R. H., and Mitchell, T. M. (2010). Toward an architecture for never-ending language learning. In *Proceedings of the Twenty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2010, Atlanta, Georgia, USA, July 11-15, 2010.*

[Carriero et al., 2019] Carriero, V. A., Gangemi, A., Mancinelli, M. L., Marinucci, L., Nuzzolese, A. G., Presutti, V., and Veninata, C. (2019). Arco: The italian cultural heritage knowledge graph. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II.*

[Chakrabarti et al., 2020] Chakrabarti, K., Chen, Z., Shakeri, S., and Cao, G. (2020). Open domain question answering using web tables. *CoRR*.

[Chang et al., 2013] Chang, K., Samdani, R., and Roth, D. (2013). A constrained latent variable model for coreference resolution. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL.

[Charton and Gagnon, 2011] Charton, E. and Gagnon, M. (2011). Poly-co: a multilayer perceptron approach for coreference detection. In *Proceedings of the Fifteenth Conference on Computational Natural Language Learning: Shared Task, CoNLL 2011, Portland, Oregon, USA, June 23-24, 2011*. ACL.

[Chen et al.,] Chen, C., Huang, H., Shiue, Y., and Chen, H. Numeral understanding in financial tweets for fine-grained crowd-based forecasting. In *2018 IEEE/WIC/ACM International Conference on Web Intelligence, WI 2018, Santiago, Chile, December 3-6, 2018*.

[Chen et al., 2017] Chen, D., Fisch, A., Weston, J., and Bordes, A. (2017). Reading wikipedia to answer open-domain questions. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers*.

[Chen et al., 2020] Chen, S., Wang, J., Jiang, F., and Lin, C. (2020). Improving entity linking by modeling latent entity type information. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press.

[Chowdhury et al., 2016] Chowdhury, S. N., Tandon, N., and Weikum, G. (2016). Know2look: Commonsense knowledge for visual search. In *Proceedings of the 5th Workshop on Automated Knowledge Base Construction, AKBC@NAACL-HLT 2016, San Diego, CA, USA, June 17, 2016*. The Association for Computer Linguistics.

[Christensen et al., 2010] Christensen, J., Soderland, S., Etzioni, O., et al. (2010). Semantic role labeling for open information extraction. In *Proceedings of the NAACL HLT 2010 first international workshop on formalisms and methodology for learning by reading*.

[Clark and Gardner, 2018] Clark, C. and Gardner, M. (2018). Simple and effective multi-paragraph reading comprehension. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers.*

[Clark and Manning, 2016] Clark, K. and Manning, C. D. (2016). Improving coreference resolution by learning entity-level distributed representations. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers.* The Association for Computer Linguistics.

[Collins and Singer, 1999] Collins, M. and Singer, Y. (1999). Unsupervised models for named entity classification. In *Joint SIGDAT Conference on Empirical Methods in Natural Language Processing and Very Large Corpora, EMNLP 1999, College Park, MD, USA, June 21-22, 1999.* Association for Computational Linguistics.

[Corro and Gemulla, 2013] Corro, L. D. and Gemulla, R. (2013). Clausie: clause-based open information extraction. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013.* International World Wide Web Conferences Steering Committee / ACM.

[Curran and Clark, 2003] Curran, J. R. and Clark, S. (2003). Language independent NER using a maximum entropy tagger. In *Proceedings of the Seventh Conference on Natural Language Learning, CoNLL 2003, Held in cooperation with HLT-NAACL 2003, Edmonton, Canada, May 31 - June 1, 2003.* ACL.

[de Sá Mesquita et al., 2013] de Sá Mesquita, F., Schmidek, J., and Barbosa, D. (2013). Effectiveness and efficiency of open relation extraction. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL.* ACL.

[Deng et al., 2013] Deng, D., Jiang, Y., Li, G., Li, J., and Yu, C. (2013). Scalable column concept determination for web tables using large knowledge bases. *Proc. VLDB Endow.*

[Denis and Baldridge, 2007] Denis, P. and Baldridge, J. (2007). Joint determination of anaphoricity and coreference resolution using integer programming. In *Human Language Technology Conference of the North American Chapter of the Association* *of Computational Linguistics, Proceedings, April 22-27, 2007, Rochester, New York, USA*. The Association for Computational Linguistics.

[Diefenbach et al., 2018] Diefenbach, D., López, V., Singh, K. D., and Maret, P. (2018). Core techniques of question answering systems over knowledge bases: a survey. *Knowl. Inf. Syst.*

[Diefenbach et al., 2019] Diefenbach, D., Migliatti, P. H., Qawasmeh, O., Lully, V., Singh, K., and Maret, P. (2019). Qanswer: A question answering prototype bridging the gap between a considerable part of the LOD cloud and end-users. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*.

[Dong et al., 2014] Dong, X., Gabrilovich, E., Heitz, G., Horn, W., Lao, N., Murphy, K., Strohmann, T., Sun, S., and Zhang, W. (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*.

[Dong et al., 2020a] Dong, X. L., Hajishirzi, H., Lockard, C., and Shiralkar, P. (2020a). 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*.

[Dong et al., 2020b] Dong, X. L., He, X., Kan, A., Li, X., Liang, Y., Ma, J., Xu, Y. E., Zhang, C., Zhao, T., Saldana, G. B., Deshpande, S., Manduca, A. M., Ren, J., Singh, S. P., Xiao, F., Chang, H., Karamanolakis, G., Mao, Y., Wang, Y., Faloutsos, C., McCallum, A., and Han, J. (2020b). Autoknow: Self-driving knowledge collection for products of thousands of types. In *KDD '20: The 26th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Virtual Event, CA, USA, August 23-27, 2020*.

[Dubey et al., 2019] Dubey, M., Banerjee, D., Abdelkawi, A., and Lehmann, J. (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*.

[Dubey et al., 2016] Dubey, M., Dasgupta, S., Sharma, A., Höffner, K., and Lehmann, J. (2016). Asknow: A framework for natural language query formalization in SPARQL. In *The Semantic Web. Latest Advances and New Domains - 13th International Conference, ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Proceedings*.

[Eberius et al., 2015] Eberius, J., Braunschweig, K., Hentsch, M., Thiele, M., Ahmadov, A., and Lehner, W. (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.*

[Efthymiou et al., 2017] Efthymiou, V., Hassanzadeh, O., Rodriguez-Muro, M., and Christophides, V. (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.*

[Elazar et al., 2019] Elazar, Y., Mahabal, A., Ramachandran, D., Bedrax-Weiss, T., and Roth, D. (2019). How large are lions? inducing distributions over quantitative attributes. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers.*

[Ensan and Bagheri, 2017] Ensan, F. and Bagheri, E. (2017). Document retrieval model through semantic linking. In *Proceedings of the Tenth ACM International Conference on Web Search and Data Mining, WSDM 2017, Cambridge, United Kingdom, February 6-10, 2017*. ACM.

[Ernst et al., 2014] Ernst, P., Meng, C., Siu, A., and Weikum, G. (2014). Knowlife: A knowledge graph for health and life sciences. In *IEEE 30th International Conference on Data Engineering, Chicago, ICDE 2014, IL, USA, March 31 - April 4, 2014.*

[Ernst et al., 2015] Ernst, P., Siu, A., and Weikum, G. (2015). Knowlife: a versatile approach for constructing a large knowledge graph for biomedical sciences. *BMC Bioinform.*

[Etzioni et al., 2004] Etzioni, O., Cafarella, M. J., Downey, D., Kok, S., Popescu, A., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2004). Web-scale information extraction in knowitall: (preliminary results). In *Proceedings of the 13th international conference on World Wide Web, WWW 2004, New York, NY, USA, May 17-20, 2004.*

[Etzioni et al., 2005] Etzioni, O., Cafarella, M. J., Downey, D., Popescu, A., Shaked, T., Soderland, S., Weld, D. S., and Yates, A. (2005). Unsupervised named-entity extraction from the web: An experimental study. *Artif. Intell.*

[Fader et al., 2011] Fader, A., Soderland, S., and Etzioni, O. (2011). Identifying relations for open information extraction. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL.

[Fang et al., 2016] Fang, W., Zhang, J., Wang, D., Chen, Z., and Li, M. (2016). Entity disambiguation by knowledge and text jointly embedding. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*. ACL.

[Fang et al., 2019] Fang, Z., Cao, Y., Li, Q., Zhang, D., Zhang, Z., and Liu, Y. (2019). Joint entity linking with deep reinforcement learning. In *The World Wide Web Conference, WWW 2019, San Francisco, CA, USA, May 13-17, 2019*. ACM.

[Ferré, 2017] Ferré, S. (2017). Sparklis: An expressive query builder for SPARQL endpoints with guidance in natural language. *Semantic Web*.

[Fetahu et al., 2019] Fetahu, B., Anand, A., and Koutraki, M. (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*.

[FitzGerald et al., 2015] FitzGerald, N., Täckström, O., Ganchev, K., and Das, D. (2015). Semantic role labeling with neural network factors. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*.

[Ganea et al., 2016] Ganea, O., Ganea, M., Lucchi, A., Eickhoff, C., and Hofmann, T. (2016). Probabilistic bag-of-hyperlinks model for entity linking. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. ACM.

[Gildea and Jurafsky, 2002] Gildea, D. and Jurafsky, D. (2002). Automatic labeling of semantic roles. *Computational Linguistics*.

[González-Conejero et al., 2018] González-Conejero, J., Casanovas, P., and Teodoro, E. (2018). Business requirements for legal knowledge graph: the LYNX platform. In *Proceedings of the 2nd Workshop on Technologies for Regulatory Compliance co-located with the 31st International Conference on Legal Knowledge and Information Systems (JURIX 2018), Groningen, The Netherlands, December 12, 2018*. CEUR-WS.org.

[Guo and Barbosa, 2014] Guo, Z. and Barbosa, D. (2014). Entity linking with a unified semantic representation. In *23rd International World Wide Web Conference, WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume*. ACM.

[Han et al., 2020] Han, X., Gao, T., Lin, Y., Peng, H., Yang, Y., Xiao, C., Liu, Z., Li, P., Zhou, J., and Sun, M. (2020). More data, more relations, more context and more openness: A review and outlook for relation extraction. In *Proceedings of the 1st Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 10th International Joint Conference on Natural Language Processing, AACL/IJCNLP 2020, Suzhou, China, December 4-7, 2020*.

[Han et al., 2011] Han, X., Sun, L., and Zhao, J. (2011). Collective entity linking in web text: a graph-based method. In *Proceeding of the 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR 2011, Beijing, China, July 25-29, 2011*. ACM.

[Hanisch et al., 2005] Hanisch, D., Fundel, K., Mevissen, H., Zimmer, R., and Fluck, J. (2005). Prominer: rule-based protein and gene entity recognition. *BMC Bioinform.*

[Haussmann et al., 2019] Haussmann, S., Seneviratne, O., Chen, Y., Ne'eman, Y., Codella, J. V., Chen, C., McGuinness, D. L., and Zaki, M. J. (2019). Foodkg: A semantics-driven knowledge graph for food recommendation. In *The Semantic Web - ISWC 2019 - 18th International Semantic Web Conference, Auckland, New Zealand, October 26-30, 2019, Proceedings, Part II.*

[He et al., 2017] He, L., Lee, K., Lewis, M., and Zettlemoyer, L. (2017). Deep semantic role labeling: What works and what's next. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics, ACL 2017, Vancouver, Canada, July 30 - August 4, Volume 1: Long Papers.*

[Heinzerling and Inui, 2021] Heinzerling, B. and Inui, K. (2021). Language models as knowledge bases: On entity representations, storage capacity, and paraphrased queries. In *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume, EACL 2021, Online, April 19 - 23, 2021.*

[Ho et al., 2019] Ho, V. T., Ibrahim, Y., Pal, K., Berberich, K., and Weikum, G. (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.*

[Ho et al., 2020] Ho, V. T., Pal, K., Kleer, N., Berberich, K., and Weikum, G. (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*.

[Ho et al., 2021a] Ho, V. T., Pal, K., Razniewski, S., Berberich, K., and Weikum, G. (2021a). Extracting contextualized quantity facts from web tables. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021*.

[Ho et al., 2021b] Ho, V. T., Pal, K., and Weikum, G. (2021b). Qute: Answering quantity queries from web tables. In *SIGMOD '21: International Conference on Management of Data, Virtual Event, China, June 20-25, 2021*.

[Ho et al., 2022] Ho, V. T., Stepanova, D., Milchevski, D., Strötgen, J., and Weikum, G. (2022). Enhancing knowledge bases with quantity facts. In *WWW '22: The ACM Web Conference 2022, Virtual Event, Lyon, France, April 25 - 29, 2022*. ACM.

[Hoffart et al., 2014] Hoffart, J., Milchevski, D., and Weikum, G. (2014). STICS: searching with strings, things, and cats. In *The 37th International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '14, Gold Coast , QLD, Australia - July 06 - 11, 2014*.

[Hoffart et al., 2011] Hoffart, J., Yosef, M. A., Bordino, I., Fürstenau, H., Pinkal, M., Spaniol, M., Taneva, B., Thater, S., and Weikum, G. (2011). Robust disambiguation of named entities in text. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL*.

[Hogan et al., 2021] Hogan, A., Blomqvist, E., Cochez, M., d'Amato, C., de Melo, G., Gutiérrez, C., Kirrane, S., Gayo, J. E. L., Navigli, R., Neumaier, S., Ngomo, A. N., Polleres, A., Rashid, S. M., Rula, A., Schmelzeisen, L., Sequeda, J. F., Staab, S., and Zimmermann, A. (2021). Knowledge graphs. *ACM Comput. Surv.*

[Hua et al., 2015] Hua, W., Zheng, K., and Zhou, X. (2015). Microblog entity linking with social temporal context. In *Proceedings of the 2015 ACM SIGMOD International Conference on Management of Data, Melbourne, Victoria, Australia, May 31 - June 4, 2015*. ACM.

[Huang et al., 2020] Huang, Z., Xu, S., Hu, M., Wang, X., Qiu, J., Fu, Y., Zhao, Y., Peng, Y., and Wang, C. (2020). Recent trends in deep learning based open-domain textual question answering systems. *IEEE Access*.

[Ibrahim et al., 2016] Ibrahim, Y., Riedewald, M., and Weikum, G. (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*.

[Ibrahim et al., 2019] Ibrahim, Y., Riedewald, M., Weikum, G., and Zeinalipour-Yazti, D. (2019). Bridging quantities in tables and text. In *35th IEEE International Conference on Data Engineering, ICDE 2019, Macao, China, April 8-11, 2019*.

[Ji et al., 2016] Ji, Z., Sun, A., Cong, G., and Han, J. (2016). Joint recognition and linking of fine-grained locations from tweets. In *Proceedings of the 25th International Conference on World Wide Web, WWW 2016, Montreal, Canada, April 11 - 15, 2016*. ACM.

[Jiang et al., 2020] Jiang, C., Nian, Z., Guo, K., Chu, S., Zhao, Y., Shen, L., and Tu, K. (2020). Learning numeral embedding. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: Findings, EMNLP 2020, Online Event, 16-20 November 2020*.

[Joshi et al., 2014] Joshi, M., Sawant, U., and Chakrabarti, S. (2014). Knowledge graph and corpus driven segmentation and answer inference for telegraphic entity-seeking queries. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*.

[Junior et al., 2020] Junior, A. C., Orlandi, F., Graux, D., Hossari, M., O'Sullivan, D., Hartz, C., and Dirschl, C. (2020). Knowledge graph-based legal search over german court cases. In *The Semantic Web: ESWC 2020 Satellite Events - ESWC 2020 Satellite Events, Heraklion, Crete, Greece, May 31 - June 4, 2020, Revised Selected Papers*. Springer.

[Kalayci et al., 2020] Kalayci, E. G., Grangel-González, I., Lösch, F., Xiao, G., ul Mehdi, A., Kharlamov, E., and Calvanese, D. (2020). Semantic integration of bosch manufacturing data using virtual knowledge graphs. In *The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part II*. Springer.

[Kantor and Globerson, 2019] Kantor, B. and Globerson, A. (2019). Coreference resolution with entity equalization. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics.

[Kazama and Torisawa, 2007] Kazama, J. and Torisawa, K. (2007). Exploiting wikipedia as external knowledge for named entity recognition. In *EMNLP-CoNLL 2007, Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, June 28-30, 2007, Prague, Czech Republic*. ACL.

[Kim and Woodland, 2000] Kim, J. and Woodland, P. C. (2000). A rule-based named entity recognition system for speech input. In *Sixth International Conference on Spoken Language Processing, ICSLP 2000 / INTERSPEECH 2000, Beijing, China, October 16-20, 2000*. ISCA.

[Kolitsas et al., 2018] Kolitsas, N., Ganea, O., and Hofmann, T. (2018). End-to-end neural entity linking. In *Proceedings of the 22nd Conference on Computational Natural Language Learning, CoNLL 2018, Brussels, Belgium, October 31 - November 1, 2018*. Association for Computational Linguistics.

[Kotnis and García-Durán, 2019] Kotnis, B. and García-Durán, A. (2019). Learning numerical attributes in knowledge bases. In *1st Conference on Automated Knowledge Base Construction, AKBC 2019, Amherst, MA, USA, May 20-22, 2019*.

[Kruit et al., 2019] Kruit, B., Boncz, P. A., and Urbani, J. (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*.

[Krupka and IsoQuest, 2005] Krupka, G. and IsoQuest, K. (2005). Description of the nerowl extractor system as used for muc-7. In *Proc. 7th Message Understanding Conf*.

[Kwiatkowski et al., 2019] Kwiatkowski, T., Palomaki, J., Redfield, O., Collins, M., Parikh, A. P., Alberti, C., Epstein, D., Polosukhin, I., Devlin, J., Lee, K., Toutanova, K., Jones, L., Kelcey, M., Chang, M., Dai, A. M., Uszkoreit, J., Le, Q., and Petrov, S. (2019). Natural questions: a benchmark for question answering research. *Trans. Assoc. Comput. Linguistics*.

[Lamm et al., 2018] Lamm, M., Chaganty, A. T., Jurafsky, D., Manning, C. D., and Liang, P. S. (2018). Qsrl : A semantic role-labeling schema for quantitative facts.

[Lassila and Swick, 1999] Lassila, O. and Swick, R. R. (1999). Resource description framework (RDF) model and syntax specification.

[Lata et al., 2022] Lata, K., Singh, P., and Dutta, K. (2022). Mention detection in coreference resolution: survey. *Applied Intelligence*.

[Le and Titov, 2018] Le, P. and Titov, I. (2018). Improving entity linking by modeling latent relations between mentions. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers*. Association for Computational Linguistics.

[Le and Titov, 2019] Le, P. and Titov, I. (2019). Distant learning for entity linking with automatic noise detection. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics.

[Lee et al., 2017] Lee, K., He, L., Lewis, M., and Zettlemoyer, L. (2017). End-to-end neural coreference resolution. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*. Association for Computational Linguistics.

[Lee et al., 2018] Lee, K., He, L., and Zettlemoyer, L. (2018). Higher-order coreference resolution with coarse-to-fine inference. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 2 (Short Papers)*.

[Lehmann et al., 2015] Lehmann, J., Isele, R., Jakob, M., Jentzsch, A., Kontokostas, D., Mendes, P. N., Hellmann, S., Morsey, M., van Kleef, P., Auer, S., and Bizer, C. (2015). Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia. *Semantic Web*.

[Lehmberg and Bizer, 2017] Lehmberg, O. and Bizer, C. (2017). Stitching web tables for improving matching quality. *Proc. VLDB Endow.*

[Lenat, 1995] Lenat, D. B. (1995). CYC: A large-scale investment in knowledge infrastructure. *Commun. ACM*.

[Li et al., 2022] Li, J., Sun, A., Han, J., and Li, C. (2022). A survey on deep learning for named entity recognition. *IEEE Trans. Knowl. Data Eng.*

[Li et al., 2020a] Li, L., Wang, P., Yan, J., Wang, Y., Li, S., Jiang, J., Sun, Z., Tang, B., Chang, T., Wang, S., and Liu, Y. (2020a). Real-world data medical knowledge graph: construction and applications. *Artif. Intell. Medicine.*

[Li et al., 2020b] Li, X., Feng, J., Meng, Y., Han, Q., Wu, F., and Li, J. (2020b). A unified MRC framework for named entity recognition. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020.* Association for Computational Linguistics.

[Li et al., 2020c] Li, X., Sun, X., Meng, Y., Liang, J., Wu, F., and Li, J. (2020c). Dice loss for data-imbalanced NLP tasks. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020.* Association for Computational Linguistics.

[Limaye et al., 2010] Limaye, G., Sarawagi, S., and Chakrabarti, S. (2010). Annotating and searching web tables using entities, types and relationships. *Proc. VLDB Endow.*

[Lin et al., 2020] Lin, B. Y., Lee, S., Khanna, R., and Ren, X. (2020). Birds have four legs?! numersense: Probing numerical commonsense knowledge of pre-trained language models. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020.*

[Liu et al., 2020a] Liu, L., Song, Z., and Zheng, X. (2020a). Improving coreference resolution by leveraging entity-centric features with graph neural networks and second-order inference. *CoRR.*

[Liu et al., 2020b] Liu, S., Sun, Y., Li, B., Wang, W., and Zhao, X. (2020b). HAMNER: headword amplified multi-span distantly supervised method for domain specific named entity recognition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020.* AAAI Press.

[Liu et al., 2011] Liu, X., Zhang, S., Wei, F., and Zhou, M. (2011). Recognizing named entities in tweets. In *The 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies, Proceedings of the Conference, 19-24 June, 2011, Portland, Oregon, USA.* The Association for Computer Linguistics.

[Liu et al., 2019a] Liu, Y., Meng, F., Zhang, J., Xu, J., Chen, Y., and Zhou, J. (2019a). GCDT: A global context enhanced deep transition architecture for sequence labeling. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics.

[Liu et al., 2019b] Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019b). Roberta: A robustly optimized BERT pretraining approach. *CoRR*.

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

[Luo et al., 2004] Luo, X., Ittycheriah, A., Jing, H., Kambhatla, N., and Roukos, S. (2004). A mention-synchronous coreference resolution algorithm based on the bell tree. In *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics, 21-26 July, 2004, Barcelona, Spain*. ACL.

[Luo et al., 2020a] Luo, X., Liu, L., Yang, Y., Bo, L., Cao, Y., Wu, J., Li, Q., Yang, K., and Zhu, K. Q. (2020a). Alicoco: Alibaba e-commerce cognitive concept net. In *Proceedings of the 2020 International Conference on Management of Data, SIGMOD Conference 2020, online conference [Portland, OR, USA], June 14-19, 2020*.

[Luo et al., 2020b] Luo, Y., Xiao, F., and Zhao, H. (2020b). Hierarchical contextualized representation for named entity recognition. In *The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020*. AAAI Press.

[Madaan et al., 2016] Madaan, A., Mittal, A., Mausam, Ramakrishnan, G., and Sarawagi, S. (2016). Numerical relation extraction with minimal supervision. In *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*.

[Martínez-Rodríguez et al., 2020] Martínez-Rodríguez, J., Hogan, A., and López-Arévalo, I. (2020). Information extraction meets the semantic web: A survey. *Semantic Web*.

[Mausam, 2016] Mausam (2016). Open information extraction systems and downstream applications. In *Proceedings of the Twenty-Fifth International Joint Conference on Artificial Intelligence, IJCAI 2016, New York, NY, USA, 9-15 July 2016*.

[Mausam et al., 2012] Mausam, Schmitz, M., Soderland, S., Bart, R., and Etzioni, O. (2012). Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*. ACL.

[McNamee and Mayfield, 2002] McNamee, P. and Mayfield, J. (2002). Entity extraction without language-specific resources. In *Proceedings of the 6th Conference on Natural Language Learning, CoNLL 2002, Held in cooperation with COLING 2002, Taipei, Taiwan, 2002*. ACL.

[Mehdi et al., 2019] Mehdi, A., Kharlamov, E., Stepanova, D., Loesch, F., and Grangel-González, I. (2019). Towards semantic integration of bosch manufacturing data. In *Proceedings of the ISWC 2019 Satellite Tracks (Posters & Demonstrations, Industry, and Outrageous Ideas) co-located with 18th International Semantic Web Conference (ISWC 2019), Auckland, New Zealand, October 26-30, 2019*. CEUR-WS.org.

[Mehta et al., 2021] Mehta, K., Oprea, I., and Rasiwasia, N. (2021). Latex-numeric: Language agnostic text attribute extraction for numeric attributes. In *Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies: Industry Papers, NAACL-HLT 2021, Online, June 6-11, 2021*.

[Meij, 2019] Meij, E. (2019). Understanding news using the bloomberg knowledge graph. *Invited talk at the Big Data Innovators Gathering (TheWebConf). Slides at https://speakerdeck. com/emeij/understanding-news-using-thebloomberg-knowledge-graph*.

[Mikheev et al., 1999] Mikheev, A., Moens, M., and Grover, C. (1999). Named entity recognition without gazetteers. In *EACL 1999, 9th Conference of the European Chapter of the Association for Computational Linguistics, June 8-12, 1999, University of Bergen, Bergen, Norway*. The Association for Computer Linguistics.

[Miller, 1995] Miller, G. A. (1995). Wordnet: A lexical database for english. *Commun. ACM*.

[Moreno et al., 2017] Moreno, J. G., Besançon, R., Beaumont, R., D'hondt, E., Ligozat, A., Rosset, S., Tannier, X., and Grau, B. (2017). Combining word and entity embeddings for entity linking. In *The Semantic Web - 14th International Conference, ESWC 2017, Portorož, Slovenia, May 28 - June 1, 2017, Proceedings, Part I.*

[Mueller and Durrett, 2018] Mueller, D. and Durrett, G. (2018). Effective use of context in noisy entity linking. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018.* Association for Computational Linguistics.

[Mulwad et al., 2013] Mulwad, V., Finin, T., and Joshi, A. (2013). Semantic message passing for generating linked data from tables. In *The Semantic Web - ISWC 2013 - 12th International Semantic Web Conference, Sydney, NSW, Australia, October 21-25, 2013, Proceedings, Part I.*

[Mulwad et al., 2010] Mulwad, V., Finin, T., Syed, Z., and Joshi, A. (2010). Using linked data to interpret tables. In *Proceedings of the First International Workshop on Consuming Linked Data, Shanghai, China, November 8, 2010.* CEUR-WS.org.

[Naik et al., 2019] Naik, A., Ravichander, A., Rosé, C. P., and Hovy, E. H. (2019). Exploring numeracy in word embeddings. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28-August 2, 2019, Volume 1: Long Papers.*

[Ng and Cardie, 2002] Ng, V. and Cardie, C. (2002). Identifying anaphoric and non-anaphoric noun phrases to improve coreference resolution. In *19th International Conference on Computational Linguistics, COLING 2002, Howard International House and Academia Sinica, Taipei, Taiwan, August 24 - September 1, 2002.*

[Nguyen et al., 2021] Nguyen, T., Razniewski, S., and Weikum, G. (2021). Advanced semantics for commonsense knowledge extraction. In *WWW '21: The Web Conference 2021, Virtual Event / Ljubljana, Slovenia, April 19-23, 2021.*

[Oulabi and Bizer, 2019] Oulabi, Y. and Bizer, C. (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.*

[Parravicini et al., 2019] Parravicini, A., Patra, R., Bartolini, D. B., and Santambrogio, M. D. (2019). Fast and accurate entity linking via graph embedding. In *Proceedings* *of the 2nd Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA), Amsterdam, The Netherlands, 30 June 2019*. ACM.

[Pennington et al., 2014] Pennington, J., Socher, R., and Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*.

[Peters et al., 2018] Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. 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)*. Association for Computational Linguistics.

[Petroni et al., 2019] Petroni, F., Rocktäschel, T., Riedel, S., Lewis, P. S. H., Bakhtin, A., Wu, Y., and Miller, A. H. (2019). Language models as knowledge bases? In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*.

[Phan et al., 2019] Phan, M. C., Sun, A., Tay, Y., Han, J., and Li, C. (2019). Pair-linking for collective entity disambiguation: Two could be better than all. *IEEE Trans. Knowl. Data Eng.*

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

[Plu et al., 2018] Plu, J., Prokofyev, R., Tonon, A., Cudré-Mauroux, P., Difallah, D. E., Troncy, R., and Rizzo, G. (2018). Sanaphor++: Combining deep neural networks with semantics for coreference resolution. In *Proceedings of the Eleventh International Conference on Language Resources and Evaluation, LREC 2018, Miyazaki, Japan, May 7-12, 2018*. European Language Resources Association (ELRA).

[Prokofyev et al., 2015] Prokofyev, R., Tonon, A., Luggen, M., Vouilloz, L., Difallah, D. E., and Cudré-Mauroux, P. (2015). SANAPHOR: ontology-based coreference resolution. In *The Semantic Web - ISWC 2015 - 14th International Semantic Web Conference, Bethlehem, PA, USA, October 11-15, 2015, Proceedings, Part I*. Springer.

[Punyakanok and Roth, 2000] Punyakanok, V. and Roth, D. (2000). The use of classifiers in sequential inference. In *Advances in Neural Information Processing Systems 13, Papers from Neural Information Processing Systems (NIPS) 2000, Denver, CO, USA*. MIT Press.

[Quimbaya et al., 2016] Quimbaya, A. P., Múnera, A. S., Rivera, R. A. G., Rodríguez, J. C. D., Velandia, O. M. M., Peña, A. A. G., and Labbé, C. (2016). Named entity recognition over electronic health records through a combined dictionary-based approach. *Procedia Computer Science*.

[Raad et al., 2020] Raad, J., Beek, W., van Harmelen, F., Wielemaker, J., Pernelle, N., and Saïs, F. (2020). Constructing and cleaning identity graphs in the LOD cloud. *Data Intell.*

[Rama-Maneiro et al., 2020] Rama-Maneiro, E., Vidal, J. C., and Lama, M. (2020). Collective disambiguation in entity linking based on topic coherence in semantic graphs. *Knowl. Based Syst.*

[Ramshaw and Marcus, 1995] Ramshaw, L. A. and Marcus, M. (1995). Text chunking using transformation-based learning. In *Third Workshop on Very Large Corpora, VLC@ACL 1995, Cambridge, Massachusetts, USA, June 30, 1995.*

[Ratinov and Roth, 2012] Ratinov, L. and Roth, D. (2012). Learning-based multi-sieve co-reference resolution with knowledge. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning, EMNLP-CoNLL 2012, July 12-14, 2012, Jeju Island, Korea*. ACL.

[Raviv et al., 2016] Raviv, H., Kurland, O., and Carmel, D. (2016). Document retrieval using entity-based language models. In *Proceedings of the 39th International ACM SIGIR conference on Research and Development in Information Retrieval, SIGIR 2016, Pisa, Italy, July 17-21, 2016*. ACM.

[Reinanda et al., 2020] Reinanda, R., Meij, E., and de Rijke, M. (2020). Knowledge graphs: An information retrieval perspective. *Found. Trends Inf. Retr.*

[Reuters, 2017] Reuters, T. (2017). Thomson reuters launches first of its kind knowledge graph feed allowing financial services customers to accelerate their ai and digital strategies. *Press Release*.

[Ritter et al., 2011] Ritter, A., Clark, S., Mausam, and Etzioni, O. (2011). Named entity recognition in tweets: An experimental study. In *Proceedings of the 2011 Conference* *on Empirical Methods in Natural Language Processing, EMNLP 2011, 27-31 July 2011, John McIntyre Conference Centre, Edinburgh, UK, A meeting of SIGDAT, a Special Interest Group of the ACL.* ACL.

[Ritze and Bizer, 2017] Ritze, D. and Bizer, C. (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.*

[Ritze et al., 2015] Ritze, D., Lehmberg, O., and Bizer, C. (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.*

[Robertson, 2004] Robertson, S. (2004). Understanding inverse document frequency: on theoretical arguments for IDF. *Journal of Documentation.*

[Rocktäschel et al., 2012] Rocktäschel, T., Weidlich, M., and Leser, U. (2012). Chemspot: a hybrid system for chemical named entity recognition. *Bioinform.*

[Romero et al., 2019] Romero, J., Razniewski, S., Pal, K., Pan, J. Z., Sakhadeo, A., and Weikum, G. (2019). Commonsense properties from query logs and question answering forums. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management, CIKM 2019, Beijing, China, November 3-7, 2019.*

[Roy and Anand, 2021] Roy, R. S. and Anand, A. (2021). *Question Answering for the Curated Web: Tasks and Methods in QA over Knowledge Bases and Text Collections.* Morgan & Claypool Publishers.

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

[Saha and Mausam, 2018] Saha, S. and Mausam (2018). Open information extraction from conjunctive sentences. In *Proceedings of the 27th International Conference on Computational Linguistics, COLING 2018, Santa Fe, New Mexico, USA, August 20-26, 2018.*

[Saha et al., 2017] Saha, S., Pal, H., 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.*

[Sandhaus, 2008] Sandhaus, E. (2008). The new york times annotated corpus.

[Sarawagi and Chakrabarti, 2014] Sarawagi, S. and Chakrabarti, S. (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*.

[Sekine and Nobata, 2004] Sekine, S. and Nobata, C. (2004). Definition, dictionaries and tagger for extended named entity hierarchy. In *Proceedings of the Fourth International Conference on Language Resources and Evaluation, LREC 2004, May 26-28, 2004, Lisbon, Portugal*. European Language Resources Association.

[Settles, 2004] Settles, B. (2004). Biomedical named entity recognition using conditional random fields and rich feature sets. In *Proceedings of the International Joint Workshop on Natural Language Processing in Biomedicine and its Applications, NLPBA/BioNLP 2004, Geneva, Switzerland, August 28-29, 2004*.

[Sevgili et al., 2019] Sevgili, Ö., Panchenko, A., and Biemann, C. (2019). Improving neural entity disambiguation with graph embeddings. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28 - August 2, 2019, Volume 2: Student Research Workshop*. Association for Computational Linguistics.

[Shaalan, 2014] Shaalan, K. (2014). A survey of arabic named entity recognition and classification. *Comput. Linguistics*.

[Shen et al., 2015] Shen, W., Wang, J., and Han, J. (2015). Entity linking with a knowledge base: Issues, techniques, and solutions. *IEEE Trans. Knowl. Data Eng.*

[Shu et al., 2017] Shu, L., Xu, H., and Liu, B. (2017). DOC: deep open classification of text documents. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing, EMNLP 2017, Copenhagen, Denmark, September 9-11, 2017*.

[Singh et al., 2018] Singh, K., Radhakrishna, A. S., Both, A., Shekarpour, S., Lytra, I., Usbeck, R., Vyas, A., Khikmatullaev, A., Punjani, D., Lange, C., Vidal, M., Lehmann, J., and Auer, S. (2018). Why reinvent the wheel: Let's build question answering systems together. In *Proceedings of the 2018 World Wide Web Conference on World Wide Web, WWW 2018, Lyon, France, April 23-27, 2018*.

[Smirnova and Cudré-Mauroux, 2019] Smirnova, A. and Cudré-Mauroux, P. (2019). Relation extraction using distant supervision: A survey. *ACM Comput. Surv.*

[Soon et al., 2001] Soon, W. M., Ng, H. T., and Lim, C. Y. (2001). A machine learning approach to coreference resolution of noun phrases. *Comput. Linguistics.*

[Speer et al., 2017] Speer, R., Chin, J., and Havasi, C. (2017). Conceptnet 5.5: An open multilingual graph of general knowledge. In *Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence, February 4-9, 2017, San Francisco, California, USA.*

[Spithourakis and Riedel, 2018] Spithourakis, G. P. and Riedel, S. (2018). Numeracy for language models: Evaluating and improving their ability to predict numbers. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics, ACL 2018, Melbourne, Australia, July 15-20, 2018, Volume 1: Long Papers.*

[Stanovsky et al., 2016] Stanovsky, G., Ficler, J., Dagan, I., and Goldberg, Y. (2016). Getting more out of syntax with props. *CoRR.*

[Strötgen and Gertz, 2016] Strötgen, J. and Gertz, M. (2016). *Domain-Sensitive Temporal Tagging.* Morgan & Claypool Publishers.

[Subramanian and Roth, 2019] Subramanian, S. and Roth, D. (2019). Improving generalization in coreference resolution via adversarial training. In *Proceedings of the Eighth Joint Conference on Lexical and Computational Semantics, *SEM@NAACL-HLT 2019, Minneapolis, MN, USA, June 6-7, 2019.* Association for Computational Linguistics.

[Suchanek et al., 2007] Suchanek, F. M., Kasneci, G., and Weikum, G. (2007). Yago: a core of semantic knowledge. In *Proceedings of the 16th International Conference on World Wide Web, WWW 2007, Banff, Alberta, Canada, May 8-12, 2007.*

[Suchanek et al., 2009] Suchanek, F. M., Sozio, M., and Weikum, G. (2009). SOFIE: a self-organizing framework for information extraction. In *Proceedings of the 18th International Conference on World Wide Web, WWW 2009, Madrid, Spain, April 20-24, 2009.*

[Sundararaman et al., 2020] Sundararaman, D., Si, S., Subramanian, V., Wang, G., Hazarika, D., and Carin, L. (2020). Methods for numeracy-preserving word embeddings. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020.*

[Szarvas et al., 2006] Szarvas, G., Farkas, R., and Kocsor, A. (2006). A multilingual named entity recognition system using boosting and C4.5 decision tree learning algorithms. In *Discovery Science, 9th International Conference, DS 2006, Barcelona, Spain, October 7-10, 2006, Proceedings*. Springer.

[Talmor and Berant, 2018] Talmor, A. and Berant, J. (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)*.

[Tanon et al., 2018] Tanon, T. P., de Assunção, M. D., Caron, E., and Suchanek, F. M. (2018). Demoing platypus - A multilingual question answering platform for wikidata. In *The Semantic Web: ESWC 2018 Satellite Events - ESWC 2018 Satellite Events, Heraklion, Crete, Greece, June 3-7, 2018, Revised Selected Papers*.

[Terolli et al., 2020] Terolli, E., Ernst, P., and Weikum, G. (2020). Focused query expansion with entity cores for patient-centric health search. In *The Semantic Web - ISWC 2020 - 19th International Semantic Web Conference, Athens, Greece, November 2-6, 2020, Proceedings, Part I*. Springer.

[Toshniwal et al., 2020] Toshniwal, S., Wiseman, S., Ettinger, A., Livescu, K., and Gimpel, K. (2020). Learning to ignore: Long document coreference with bounded memory neural networks. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics.

[Tran et al., 2015] Tran, A. T., Tran, N. K., Hadgu, A. T., and Jäschke, R. (2015). Semantic annotation for microblog topics using wikipedia temporal information. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*. The Association for Computational Linguistics.

[Unger et al., 2015] Unger, C., Forascu, C., López, V., Ngomo, A. N., Cabrio, E., Cimiano, P., and Walter, S. (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*.

[Unger et al., 2014] Unger, C., Freitas, A., and Cimiano, P. (2014). An introduction to question answering over linked data. In *Reasoning Web. Reasoning on the Web in the*

Big Data Era - 10th International Summer School 2014, Athens, Greece, September 8-13, 2014. Proceedings.

[Unger et al., 2016] Unger, C., Ngomo, A. N., and Cabrio, E. (2016). 6th open challenge on question answering over linked data (QALD-6). In *Semantic Web Challenges - Third SemWebEval Challenge at ESWC 2016, Heraklion, Crete, Greece, May 29 - June 2, 2016, Revised Selected Papers.*

[Usbeck et al., 2019] Usbeck, R., Röder, M., Hoffmann, M., Conrads, F., Huthmann, J., Ngomo, A. N., Demmler, C., and Unger, C. (2019). Benchmarking question answering systems. *Semantic Web.*

[Venetis et al., 2011] Venetis, P., Halevy, A. Y., Madhavan, J., Pasca, M., Shen, W., Wu, F., Miao, G., and Wu, C. (2011). Recovering semantics of tables on the web. *Proc. VLDB Endow.*

[Vrandecic and Krötzsch, 2014] Vrandecic, D. and Krötzsch, M. (2014). Wikidata: a free collaborative knowledge base. *Commun. ACM.*

[Wallace et al., 2019] Wallace, E., Wang, Y., Li, S., Singh, S., and Gardner, M. (2019). Do NLP models know numbers? probing numeracy in embeddings. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019.*

[Wan et al., 2019] Wan, H., Zhang, Y., Zhang, J., and Tang, J. (2019). Aminer: Search and mining of academic social networks. *Data Intell.*

[Wang et al., 2017] Wang, Q., Mao, Z., Wang, B., and Guo, L. (2017). Knowledge graph embedding: A survey of approaches and applications. *IEEE Trans. Knowl. Data Eng.*

[Wei et al., 2019] Wei, F., Nguyen, U. T., and Jiang, H. (2019). Dual-fofe-net neural models for entity linking with pagerank. In *Artificial Neural Networks and Machine Learning - ICANN 2019 - 28th International Conference on Artificial Neural Networks, Munich, Germany, September 17-19, 2019, Proceedings - Workshop and Special Sessions.* Springer.

[Weikum, 2020] Weikum, G. (2020). Entities with quantities. *IEEE Data Eng. Bull.*

[Weikum et al., 2021] Weikum, G., Dong, X. L., Razniewski, S., and Suchanek, F. M. (2021). Machine knowledge: Creation and curation of comprehensive knowledge bases. *Found. Trends Databases.*

[Wiseman et al., 2016] Wiseman, S., Rush, A. M., and Shieber, S. M. (2016). Learning global features for coreference resolution. In *NAACL HLT 2016, The 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego California, USA, June 12-17, 2016*. The Association for Computational Linguistics.

[Wiseman et al., 2015] Wiseman, S., Rush, A. M., Shieber, S. M., and Weston, J. (2015). Learning anaphoricity and antecedent ranking features for coreference resolution. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*. The Association for Computer Linguistics.

[Wu and Weld, 2010] Wu, F. and Weld, D. S. (2010). Open information extraction using wikipedia. In *ACL 2010, Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics, July 11-16, 2010, Uppsala, Sweden*. The Association for Computer Linguistics.

[Wu et al., 2020a] Wu, L., Petroni, F., Josifoski, M., Riedel, S., and Zettlemoyer, L. (2020a). Scalable zero-shot entity linking with dense entity retrieval. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing, EMNLP 2020, Online, November 16-20, 2020*. Association for Computational Linguistics.

[Wu et al., 2016] Wu, T., Yan, S., Piao, Z., Xu, L., Wang, R., and Qi, G. (2016). Entity linking in web tables with multiple linked knowledge bases. In *Semantic Technology - 6th Joint International Conference, JIST 2016, Singapore, Singapore, November 2-4, 2016, Revised Selected Papers*. Springer.

[Wu et al., 2020b] Wu, W., Wang, F., Yuan, A., Wu, F., and Li, J. (2020b). Corefqa: Coreference resolution as query-based span prediction. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, ACL 2020, Online, July 5-10, 2020*. Association for Computational Linguistics.

[Wu and Palmer, 1994] Wu, Z. and Palmer, M. S. (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*.

[Xia et al., 2019] Xia, C., Zhang, C., Yang, T., Li, Y., Du, N., Wu, X., Fan, W., Ma, F., and Yu, P. S. (2019). Multi-grained named entity recognition. In *Proceedings* *of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*. Association for Computational Linguistics.

[Xia et al., 2020] Xia, P., Sedoc, J., and Durme, B. V. (2020). Revisiting memory-efficient incremental coreference resolution. *CoRR*.

[Xu et al., 2016] Xu, K., Reddy, S., Feng, Y., Huang, S., and Zhao, D. (2016). Question answering on freebase via relation extraction and textual evidence. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, ACL 2016, August 7-12, 2016, Berlin, Germany, Volume 1: Long Papers*.

[Yagnik and Islam, 2007] Yagnik, J. and Islam, A. (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*.

[Yahya et al., 2014] Yahya, M., Whang, S., Gupta, R., and Halevy, A. Y. (2014). Renoun: Fact extraction for nominal attributes. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*. ACL.

[Yakout et al., 2012] Yakout, M., Ganjam, K., Chakrabarti, K., and Chaudhuri, S. (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*.

[Yamada et al., 2016] Yamada, I., Shindo, H., Takeda, H., and Takefuji, Y. (2016). Joint learning of the embedding of words and entities for named entity disambiguation. In *Proceedings of the 20th SIGNLL Conference on Computational Natural Language Learning, CoNLL 2016, Berlin, Germany, August 11-12, 2016*. ACL.

[Yang et al., 2019] Yang, X., Gu, X., Lin, S., Tang, S., Zhuang, Y., Wu, F., Chen, Z., Hu, G., and Ren, X. (2019). Learning dynamic context augmentation for global entity linking. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing, EMNLP-IJCNLP 2019, Hong Kong, China, November 3-7, 2019*. Association for Computational Linguistics.

[Yang et al., 2004] Yang, X., Su, J., Zhou, G., and Tan, C. L. (2004). An np-cluster based approach to coreference resolution. In *COLING 2004, 20th International Conference on Computational Linguistics, Proceedings of the Conference, 23-27 August 2004, Geneva, Switzerland.*

[Yang et al., 2003] Yang, X., Zhou, G., Su, J., and Tan, C. L. (2003). Coreference resolution using competition learning approach. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics, 7-12 July 2003, Sapporo Convention Center, Sapporo, Japan.* ACL.

[Yang et al., 2018] Yang, Z., Qi, P., Zhang, S., Bengio, Y., Cohen, W. W., Salakhutdinov, R., and Manning, C. D. (2018). Hotpotqa: A dataset for diverse, explainable multi-hop question answering. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing, Brussels, Belgium, October 31 - November 4, 2018.*

[Yih et al., 2015] Yih, W., Chang, M., He, X., and Gao, J. (2015). Semantic parsing via staged query graph generation: Question answering with knowledge base. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers.*

[Zhai, 2008] Zhai, C. (2008). Statistical language models for information retrieval: A critical review. *Foundations and Trends in Information Retrieval.*

[Zhang et al., 2020] Zhang, H., Liu, X., Pan, H., Song, Y., and Leung, C. W. (2020). ASER: A large-scale eventuality knowledge graph. In *WWW '20: The Web Conference 2020, Taipei, Taiwan, April 20-24, 2020.*

[Zhang and Balog, 2020] Zhang, S. and Balog, K. (2020). Web table extraction, retrieval, and augmentation: A survey. *ACM Trans. Intell. Syst. Technol.*

[Zhang, 2017] Zhang, Z. (2017). Effective and efficient semantic table interpretation using tableminer^{+}. *Semantic Web.*

[Zheng et al., 2018] Zheng, W., Yu, J. X., Zou, L., and Cheng, H. (2018). Question answering over knowledge graphs: Question understanding via template decomposition. *PVLDB.*

[Zhou and Su, 2002] Zhou, G. and Su, J. (2002). Named entity recognition using an hmm-based chunk tagger. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics, July 6-12, 2002, Philadelphia, PA, USA*. ACL.

[Zhou and Xu, 2015] Zhou, J. and Xu, W. (2015). End-to-end learning of semantic role labeling using recurrent neural networks. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing of the Asian Federation of Natural Language Processing, ACL 2015, July 26-31, 2015, Beijing, China, Volume 1: Long Papers*.

[Zhu et al., 2005] Zhu, J., Uren, V. S., and Motta, E. (2005). Espotter: Adaptive named entity recognition for web browsing. In *Professional Knowledge Management, Third Biennial Conference, WM 2005, Kaiserslautern, Germany, April 10-13, 2005, Revised Selected Papers*. Springer.