Knowledge Graphs & Entity Linking

(and their relation to LLMs)

Session 1 of the SIGIR'23 Tutorial
"Neuro-Symbolic Representations for IR"
Taipei, Taiwan, July 23, 2023

Hannah Bast
Chair of Algorithms and Data Structures
University of Freiburg, Germany

Knowledge Graphs 1/8

- Resource Description Framework
  - RDF is based on a strikingly simple principle: ALL DATA, whatever it is, is modelled as a **set of triples**

    subject predicate object

  - For example:

<Taipei>	<capital of>	<Taiwan>
<Taipei>	<coordinates>	"25°02'N 121°34'E"
<Taiwan>	<language>	<Hokkien>
<Taiwan>	<language>	<Hakka>
<Taiwan>	<language>	<Mandarin>

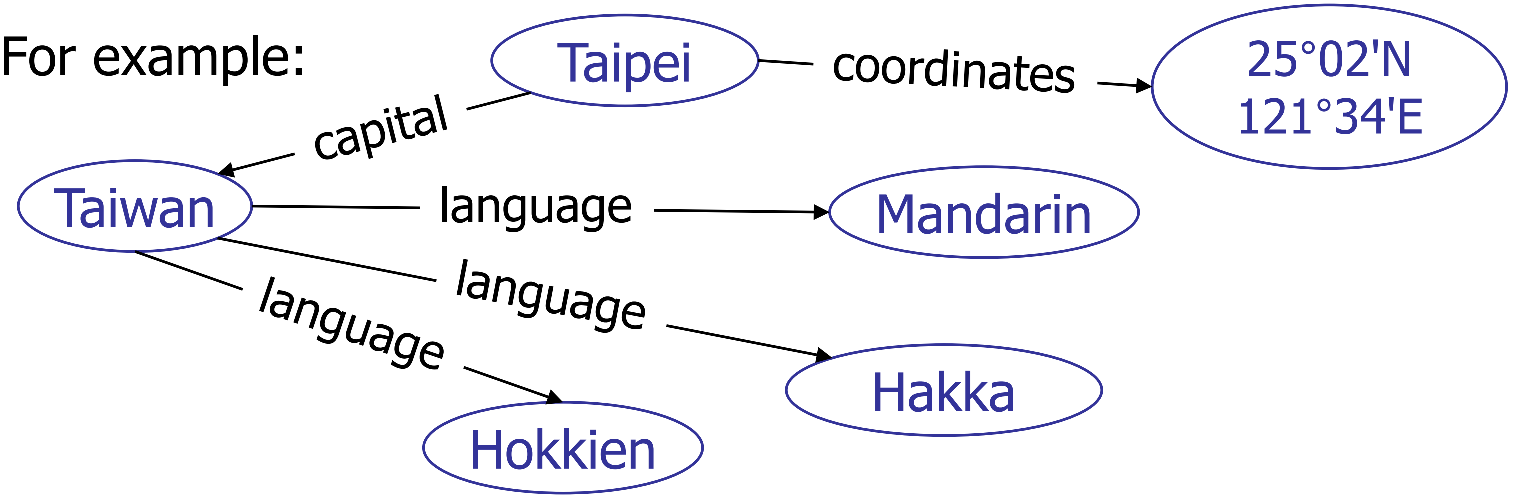
Graph view: each distinct subject or object is a **node**, each triple is a **directed edge** from subject to object, with the predicate as label

Knowledge Graphs 1/8

- Resource Description Framework
  - RDF is based on a strikingly simple principle: ALL DATA, whatever it is, is modelled as a **set of triples**

    subject predicate object

  - For example:



Graph view: each distinct subject or object is a **node**, each triple is a **directed edge** from subject to object, with the predicate as label

Knowledge Graphs 2/8

wdt:P17 ~ country
wd:Q865 ~ Taiwan
wdt:P625 ~ coordinates

UNI FREIBURG

- International Resource Identifier (IRI)
  - Names have globally unique identifiers, so-called IRIs

    An IRI is like an address in a browser (URL), only that Unicode characters like ä, ö, ü, … are allowed
  - The IRI is often not human-readable but ID-like; instead there are dedicated triples for its names and aliases
  - For example, the Wikidata IRI for Taipei is

    https://www.wikidata.org/wiki/Q1867

    And some example triples (with **IRI prefixes**) look like this:

    wd:Q1867 **rdfs:**label "Taipeh"@de
    wd:Q1867 **wdt:**P17 **wd:**Q865
    wd:Q1867 **wdt:**P625 "Point(25.03 121.57)"^^**geo:**wktLiteral

Knowledge Graphs 3/8

- Some widely used knowledge graphs
  - **Wikidata:** Started 2013, successor of Freebase (bought by Google in 2010 for 99M $), crowd-sourced, amazing coverage

    18 **B** triples, 52 131 predicates
  - **UniProt:** Started 2002, protein sequences, genes, all kinds of metadata, …

    110 **B** triples, 310 predicates
  - **PubChem:** Started 2004, chemical compounds, substances, proteins, genes + how this is all interrelated

    124 **B** triples, 426 predicates

Knowledge Graphs 4/8

- And more …
  - **OpenStreetMap:** Started 2004, all the geo-data of the world, crowd-sourced, amazing coverage and quality

    14 **B** triples, 92 704 predicates

  - **DBLP:** Started 1993, publication meta-data for computer science and adjacent fields

    0.8 **B** triples, 75 predicates

  - Most of the IT-heavy companies maintain their own (very large) knowledge graph:

    Google, Amazon, Microsoft, Meta, Bloomberg, Walmart, Airbnb, …

Knowledge Graphs 5/8

- Historical knowledge graphs
  - **Freebase:** Started 2007, acquired by Google in 2010, closed down in 2015

    3 B triples, 784 977 predicates
  - **YAGO:** Started 2008, derived from Wikipedia info-boxes and WordNet

    0.1 B triples, 100 predicates
  - **DBpedia**: all kinds of structured data extracted from Wikipedia

    0.8 B triples, 54 780 predicates

Al three are still mentioned and used in papers (because old benchmarks use them and academia is very slow to adapt)

Knowledge Graphs 6/8

- The standard query language for RDF is SPARQL
  - SPARQL is a variant of SQL, adapted to the (simple) RDF

```
SELECT ?title ?author ?year WHERE {
  ?paper dblp:title ?title .
  ?paper dblp:authoredBy ?author.
  ?paper dblp:yearOfPublication ?year .
  FILTER (?year <= 1940)
}
```

All papers in DBLP published before 1940

Query on QLever

  - The result of a SPARQL query is always a **table**, in the example:

    All assignments to ?title ?author ?year such that the triples exist in the knowledge graph and the FILTER condition is true

    Note: if a paper has **k** authors, there will be **k** rows for it

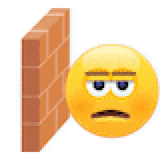
Knowledge Graphs 7/8

This allows "federated" SPARQL queries like

all power lines in the EU

- Interoperability
  - One of the strengths of RDF and SPARQL is the great ease, with which different datasets can be **combined**

    For standard databases this is a nightmare: different and often complex schemas, different identifiers, etc.



    In RDF, all you need are additional triples relating the IDs from the two datasets you want to combine

wd:Q183 **wdtn:**P402 **osmrel:**51477 [in Wikidata]

osmrel:51477 **osm**:wikidata **wd:**Q183 [in OpenStreetMap]

wd:Q183	IRI for Germany in Wikidata
osmrel:51477	IRI for Germany in OpenStreetMap
wdtn:P402	predicate for "OpenStreetMap IRI" in Wikidata
osm:wikidata	predicate for "Wikidata IRI" in OpenStreetMap

Knowledge Graphs 8/8

- A variety of example queries
  - Birth places of people with first name X — Wikidata
  - Notable events that happened on July 23 — Wikidata
  - Organisms with their proteins and sequences — UniProt
  - NSAID drugs with small molecular weight — PubChem
  - All streets in X — OpenStreetMap
  - All countries with official language X — Wikidata+OSM
  - Average number of authors per year — DBLP

Side note: These are all running with the same system, with zero configuration per dataset, that is the power of RDF+SPARQL

Finding the right SPARQL query is hard 1/4

- Example question
  - Consider the following simple search request

    Which Oscars did Meryl Streep win and for which movies?

  - The result we are looking for is something like this:

Academy Award for Best Supporting Actress	Kramer vs. Kramer
Academy Award for Best Actress	Sophie's Choice
Academy Award for Best Actress	The Iron Lady

  - On the next slide, you see the correct SPARQL query on the Wikidata knowledge graph

Finding the right SPARQL query is hard 2/4

```
PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>
PREFIX wdt:  <http://www.wikidata.org/prop/direct/>
PREFIX pq:   <http://www.wikidata.org/prop/qualifier/>
PREFIX ps:   <http://www.wikidata.org/prop/statement/>
PREFIX p:    <http://www.wikidata.org/prop/>
PREFIX wd:   <http://www.wikidata.org/entity/>
SELECT ?movie ?award WHERE {
  wd:Q873     p:P166     ?mediator .
  ?mediator   ps:P166    ?award_id .
  ?mediator   pq:P1686   ?movie_id .
  ?award_id   wdt:P31    wd:Q19020 .
  ?award_id   rdfs:label ?award . FILTER (LANG(?award) = "en")
  ?movie_id   rdfs:label ?movie . FILTER (LANG(?movie) = "en")
}
```

Finding the right SPARQL query is hard 3/4

- What's hard about finding this query?
  - Knowing the right prefix definitions

    PREFIX rdfs: <http://www.w3.org/2000/01/rdf-schema#>

  - Knowing the right entity names

    wd:Q873 ("Meryl Streep"), wd:Q19020 ("Academy Award")

  - Knowing the right predicate names **very hard, even for experts**

p:P166	"won award"	from awardee to mediator
ps:P166	"won award"	from mediator to award entity
pq:P1686	"for work"	from mediator to movie
wdt:P31	"instance of"	from "instance" to "class"

  - Knowing the syntax for filtering by language

    FILTER (LANG(?award) = "en")

Finding the right SPARQL query is hard 4/4

- A technical solution: query building tools
  - **Tool 1:** Wikidata's own query builder

    Simple autocompletion, not context-sensitive

    Requires **deep expert knowledge** of the prefixes and predicate names → limited usefulness for complex queries
  - **Tool 2:** QLever's query builder

    Advanced autocompletion, context-sensitive + as you type

    Requires only relatively **basic knowledge** of RDF/SPARQL, and little or no knowledge about the dataset

    Fun fact: the suggestions are themselves computed via SPARQL queries (on the same knowledge graph)

Knowledge Graphs and LLMs 1/3

UNI FREIBURG

- Ask questions directly in natural language
  - Let's try the following question on ChatGPT

    Which Oscars did Meryl Streep win and for which movies?

    Alas, it works like a charm

    So does this mean that we don't need knowledge graphs any longer, but we can just ask our LLM anything?

Knowledge Graphs and LLMs 2/3

- Ask questions directly in natural language
  - Let's try a slightly more difficult question on ChatGPT

    Birth place coordinates of people named X in Wikipedia

    Provides some examples, but no coordinates, says that it has no access to the whole Wikipedia

    Of course, a LLM like GPT has seen the whole Wikipedia, still it is no substitute for a database-like engine because:

    1. Storing **a lot of detail** information **accurately** in a LLM is possible, but inefficient (think of the human brain)
    2. Some queries (see slide 10) require a large amount of **working memory**, which LLMs do not have by design

Knowledge Graphs and LLMs 3/3

- Use LLMs to help formulate SPARQL queries
  - Let's repeat the query from the previous slides with a twist

    Birth place coordinates of people named X in Wikidata

    ChatGPT says it can't do it, but suggests a SPARQL query

    That query is almost right, except the first name IRI

    Again, it's hard for a LLM to store large amounts of detail information (in this case: which entity has which IRI in Wikidata)

    If we give ChatGPT a hint about the right IRI, it gives us the correct query

Entity Linking 1/8

- Definition
  - Entity Recognition (ER): identify passages in a text that refer to an entity from a given knowledge graph
  - Entity Disambiguation (ED): identify, exactly which entity the passage refers to
  - Entity Linking (EL): Entity Recognition + Disambiguation

    American athlete Whittington failed to appear in the 2013–14 season due to a torn ACL.

    American [Q30] athlete **Whittington** [Q21066526] failed to appear in the **2013–14 season** [Q16192072] due to a **torn ACL** [Q18912826].

Entity Linking 2/8

- Research on Entity Linking
  - There are thousands of research papers on the problem

    Top venues ER   Top venues ED   Top venus EL

  - Most of these papers claim to improve on previous work and have an evaluation to prove it
  - But when you actually try these linkers in practice, you frequently make one of the following three experiences:

    1. There is no software or it does not run (anymore)
    2. Lots of hyperparameters and results cannot be replicated
    3. Poor results on datasets not evaluated in the paper

    This is the norm and not the exception. **Why is that?**

Entity Linking 3/8

- Reason 1: Coarse evaluation metrics
  - The typical evaluation measures **precision**, **recall**, **F1**

    Fine to get a quick or initial impression of an approach

    But aggregate measures invite **overfitting**, especially for benchmarks that have been around for a long time
  - Here are two absolute **musts** for practically useful work on entity recognition / disambiguation / linking

    1. **Look at individual results** … in order to check whether the benchmark and the results make sense
    2. **Conduct a detailed error analysis** … in order to understand where mistakes are made and why

    It is amazing how much research work does **not** do this

Entity Linking 4/8

- Reason 2: Benchmarks artifacts and biases
  - Here are the most frequent ones in existing benchmarks:

    1. Almost exclusive **focus on named entities**

    Because it's well-defined and easy to recognize (Capitalization), but many interesting entities are in lower case

    2. **What else should count as an entity?**

    No so easy: for example, if everything is an entity that has a Wikipedia article, then almost each piece of text qualifies

    3. There often is **ambiguity what the right entity is**

    For example, being "American" → [Q30] or [Q7976] or [Q846570] ?

    4. Many benchmarks have a **bias towards certain entities**

    For example, AIDA-CoNNL: many entities are sports teams

Entity Linking 5/8

- A general-purpose analysis tool
  - **ELEVANT**: **E**ntity **L**inking **Ev**aluation and **An**alysis **T**ool

    Canonical user-friendly UI to examine individual results
    (the kind of tool everybody needs, but nobody builds … so far)

    Automatic error analysis of any given entity linker

    Let's look at a demo: https://elevant.cs.uni-freiburg.de

    When you develop your own entity linker, it is highly recommended that you use this to inspect your results

    You can run it yourself (for any linkers or benchmarks you are interested in) or examine the results on the demo page

Entity Linking 6/8

- Fair comparison of existing linkers
  - New study on which linkers **actually** work well (or not)

    A Fair [...] Evaluation of Existing End-to-End Entity Linkers
    Bast, Hertel, Prange: https://arxiv.org/abs/2305.14937
  - Compares the best or most well-known linkers in-depth (on existing benchmarks as well as on two new benchmarks)

    ReFinED, REL, GENRE, Ambiverse, TagMe, ..., **Baseline**
  - Points out problems with widely used existing benchmarks

    AIDA-CoNNL, KORE50, MSNBC, DBpedia Spotlight, ...

Demo: Compare four linkers on AIDA-CoNNL vs. News-Fair

Demo: Metonyms of Ambiverse vs. Baseline on AIDA-CoNNL

Entity Linking 7/8

- Two new fair benchmarks
  - **Wiki-Fair:** random sentences from random Wikipedia articles with 1482 ground truth annotations
  - **News-Fair:** random articles from a web-news crawl with 359 ground truth annotations
  - What is special about these benchmarks:

    1. Annotated with Wikidata IDs (standard so far: Wikipedia URLs)
    2. Alternative annotations when ER or ED is ambiguous
    3. Well defined annotations via a set of types (that cover all entities annotated in existing benchmarks)

       That way, you can easily include only those types you are interested in for your particular application or evaluation

Entity Linking 8/8

- Entity Linking and LLMs
  - With the right prompt, LLMs can also perform entity linking

    Let's try it live on ChatGPT (with the GPT-4 model)
  - Observations

    1. It works surprisingly well right out of the box
    2. To obtain great results, significant fine-tuning is needed
    3. In particular, the models get most of the rare QIDs wrong
    4. Entity linking using an LLM is **expensive** and **slow**

    If you want high-quality results with high performance, more tailored solutions are currently still the way to go

References

- Knowledge Graphs
  - https://query.wikidata.org
  - https://github.com/ad-freiburg/qlever
  - https://qlever.cs.uni-freiburg.de
  - https://aqqu.cs.uni-freiburg.de
- Entity Linking
  - https://elevant.cs.uni-freiburg.de
  - https://arxiv.org/abs/2305.14937