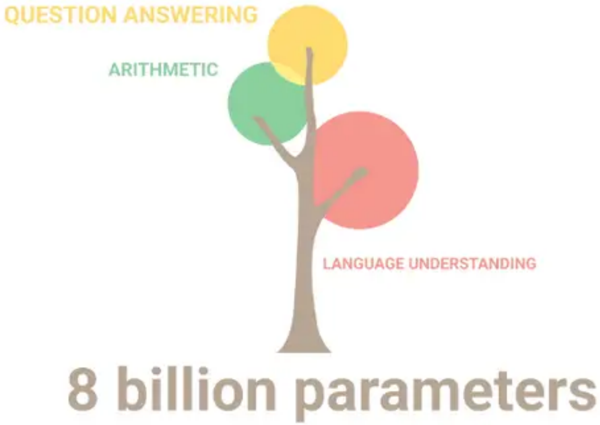
generated by Midjourney AI

Fully Automatic?

generated by Midjourney AI

LLM Task Areas



https://lastweekin.ai/p/multi-modal-ai

Blog post from Jacky Liang May 01, 2022

Related Work LLM-KG

- **2022 Narayan et al.** “Can Foundation Models Wrangle Your Data?” instruction prompts are used with large foundation models (released before the 2023 era, like GPT3) to perform entity matching, error detection, schema matching, data transformation, and data imputation tasks
- **2023 Zhu et al.** ”Llms for knowledge graph construction and reasoning: Recent capabilities and future opportunities” investigated the performance of LLMs for KGC w.r.t. entity, relation, and event extraction as well as link prediction, on eight benchmark datasets
- **2023 SPIRES** recursively performs prompt interrogation to directly extract triples from text matching either a provided LinkML schema or identifiers from existing ontologies and vocabularies
- **2023 Olala** feeds textual descriptions of ontology candidate members into an LLM to perform binary or multiple-choice ontology matching decisions.

Related Work LLM-KG

- **2023 AutoAlign** uses LLMs to refine ontology/vocabulary mappings
- **2023 Arora et al.** *“Language models enable simple systems for generating structured views of heterogeneous data lakes“* A method that generates code using LLMs to create views on heterogeneous data lakes
- **2024 TechGPT-2.0** is a model trained specifically for KGC tasks, including named entity recognition and relationship triple extraction
- **2023 Frey et al.** “Benchmarking the abilities of large language models for RDF knowledge graph creation and comprehension: How well do llms speak turtle?”, “Assessing the evolution of llm capabilities for knowledge graph engineering in 2023” investigates KG engineering tasks, RDF querying, and generation

Related Work LLM-KG

- **2023 AutoAlign** uses LLMs to refine ontology/vocabulary mappings
- **2023 Arora et al.** *“Language models enable simple systems for generating structured views of heterogeneous data lakes“* A method that generates code using LLMs to create views on heterogeneous data lakes
- **2024 TechGPT-2.0** is a model trained specifically for KGC tasks, including named entity recognition and relationship triple extraction
- **2023 Frey et al.** “Benchmarking the abilities of large language models for RDF knowledge graph creation and comprehension: How well do llms speak turtle?”, “Assessing the evolution of llm capabilities for knowledge graph engineering in 2023” investigates KG engineering tasks, RDF querying, and generation

costly: resources, time, money ⇒ configure existing tools (interfaces)

Related Work LLM-KG

- **2023 AutoAlign** uses LLMs to refine ontology/vocabulary mappings
- **2023 Arora et al.** *“Language models enable simple systems for generating structured views of heterogeneous data lakes“* A method that generates code using LLMs to create views on heterogeneous data lakes
- **2024 TechGPT-2.0** is a model trained specifically for KGC tasks, including named entity recognition and relationship triple extraction
- **2023 Frey et al.** “Benchmarking the abilities of large language models for RDF knowledge graph creation and comprehension: How well do llms speak turtle?”, “Assessing the evolution of llm capabilities for knowledge graph engineering in 2023” investigates KG engineering tasks, RDF querying, and generation

costly: resources, time, money ⇒ configure existing tools (interfaces)

- **2024 Ontogenix & R2[RML]-ChatGPT** recent effort to generate ontologies and RML mappings

Towards self-configuring Knowledge Graph Construction - A Case Study with RML

Marvin Hofer, Johannes Frey, Erhard Rahm

GEFÖRDERT VOM

Bundesministerium
für Bildung
und Forschung

SACHSEN

Diese Maßnahme wird gefördert durch die Bundesregierung aufgrund eines Beschlusses des Deutschen Bundestages. Diese Maßnahme wird mitfinanziert durch Steuermittel auf der Grundlage des von den Abgeordneten des Sächsischen Landtags beschlossenen Haushaltes.

Experiment Method and Setup

1. Test Data & Prompt Input Data
2. Target Ontology
3. RML Mapping Requirements & Challenges
4. LLM Instructions

Finding a fitting Domain and Data

Test Data & Prompt Input Data

- **IMDB movie data**, describes **films** (creative works) and **involved people**
- Includes **properties** for films like *year, name, genre, episode*
- and **relations** to persons (job categories): *actor, writer, editor, producer, director*
- Available as **6 CSV dumps** https://developer.imdb.com/non-commercial-datasets/

Test Data & Prompt Input Data

- **IMDB movie data**, describes **films** (creative works) and **involved people**
- Includes **properties** for films like *year, name, genre, episode*
- and **relations** to persons (job categories): *actor, writer, editor, producer, director*
- Available as **6 CSV dumps** https://developer.imdb.com/non-commercial-datasets/

- Representing the data for a single film and related people as **single JSON object instead of multiple tables**
  - wide table is difficult and introduces cell redundancy
  - supports simple **datatypes (numbers, strings, booleans)**

Test Data & Prompt Input Data

- **IMDB movie data**, describes **films** (creative works) and **involved people**
- Includes **properties** for films like *year, name, genre, episode*
- and **relations** to persons (job categories): *actor, writer, editor, producer, director*
- Available as **6 CSV dumps** https://developer.imdb.com/non-commercial-datasets/

- Representing the data for a single film and related people as **single JSON object instead of multiple tables**
  - wide table is difficult and introduces cell redundancy
  - supports simple **datatypes (numbers, strings, booleans)**

choosing one film entry “Diamonds” covers all job categories

Snippet of Input Data and Expected RDF Graph

```
{
 "id": "tt0167423",
 "originalTitle" : "Diamonds",
 "runtimeMinutes" : 91,
 "startYear" : 1999,
 "genre" : ["Comedy","Mistery"],
 "titleTyp" : "movie",
 "isAdult" : 0,
 "involvedPeople" : [{
   "id" : "nm0000018",
   "ordering" : 1,
   "name" : "Kirk Douglas",
   "birthYear" : 1916,
   "deathYear" : 2020,
   "category" : "actor" }, ...]
}
```

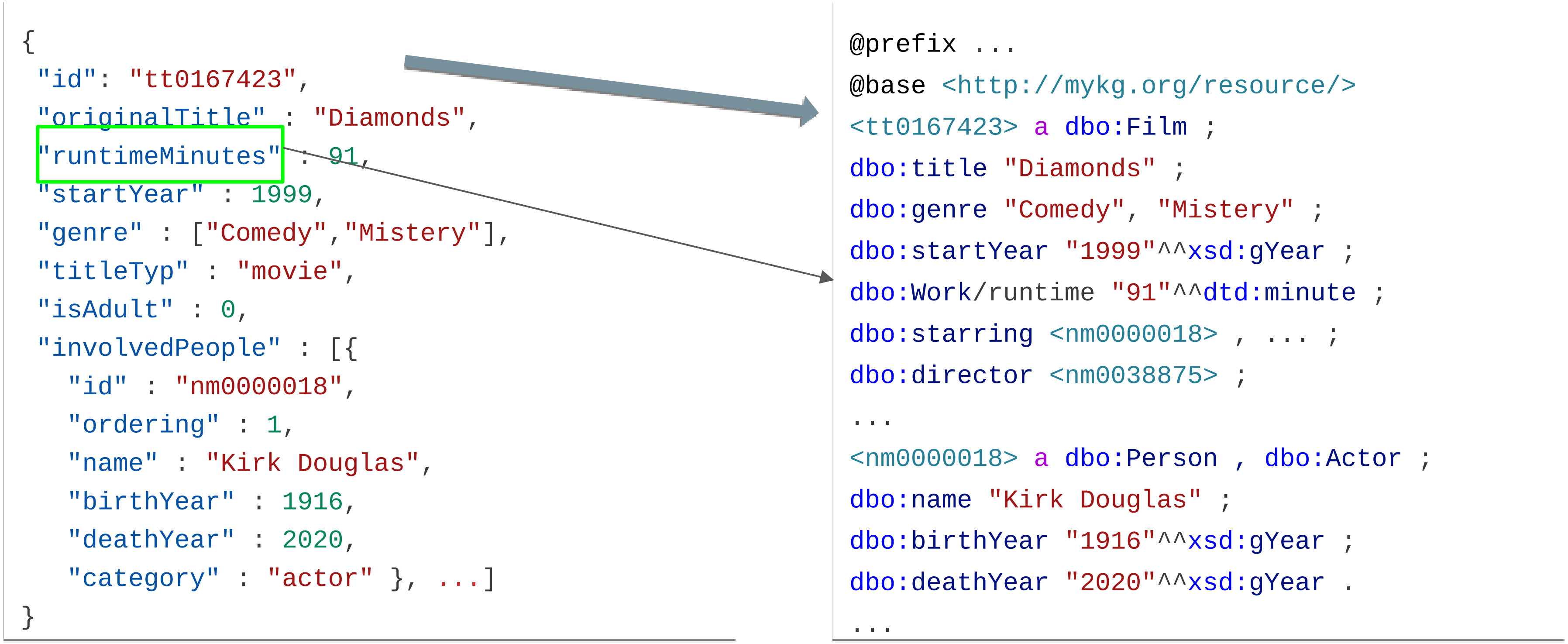
```
@prefix ...
@base <http://mykg.org/resource/>
<tt0167423> a dbo:Film ;
dbo:title "Diamonds" ;
dbo:genre "Comedy", "Mistery" ;
dbo:startYear "1999"^^xsd:gYear ;
dbo:Work/runtime "91"^^dtd:minute ;
dbo:starring <nm0000018> , ... ;
dbo:director <nm0038875> ;
...
<nm0000018> a dbo:Person , dbo:Actor ;
dbo:name "Kirk Douglas" ;
dbo:birthYear "1916"^^xsd:gYear ;
dbo:deathYear "2020"^^xsd:gYear .
...
```

Snippet of Input Data and Expected RDF Graph

```
{
 "id": "tt0167423",
 "originalTitle" : "Diamonds",
 "runtimeMinutes" : 91,
 "startYear" : 1999,
 "genre" : ["Comedy","Mistery"],
 "titleTyp" : "movie",
 "isAdult" : 0,
 "involvedPeople" : [{
   "id" : "nm0000018",
   "ordering" : 1,
   "name" : "Kirk Douglas",
   "birthYear" : 1916,
   "deathYear" : 2020,
   "category" : "actor" }, ...]
}
```

```
@prefix ...
@base <http://mykg.org/resource/>
<tt0167423> a dbo:Film ;
dbo:title "Diamonds" ;
dbo:genre "Comedy", "Mistery" ;
dbo:startYear "1999"^^xsd:gYear ;
dbo:Work/runtime "91"^^dtd:minute ;
dbo:starring <nm0000018> , ... ;
dbo:director <nm0038875> ;
...
<nm0000018> a dbo:Person , dbo:Actor ;
dbo:name "Kirk Douglas" ;
dbo:birthYear "1916"^^xsd:gYear ;
dbo:deathYear "2020"^^xsd:gYear .
...
```

Snippet of Input Data and Expected RDF Graph



```
{
 "id": "tt0167423",
 "originalTitle" : "Diamonds",
 "runtimeMinutes" : 91,
 "startYear" : 1999,
 "genre" : ["Comedy","Mistery"],
 "titleTyp" : "movie",
 "isAdult" : 0,
 "involvedPeople" : [{
   "id" : "nm0000018",
   "ordering" : 1,
   "name" : "Kirk Douglas",
   "birthYear" : 1916,
   "deathYear" : 2020,
   "category" : "actor" }, ...]
}
```

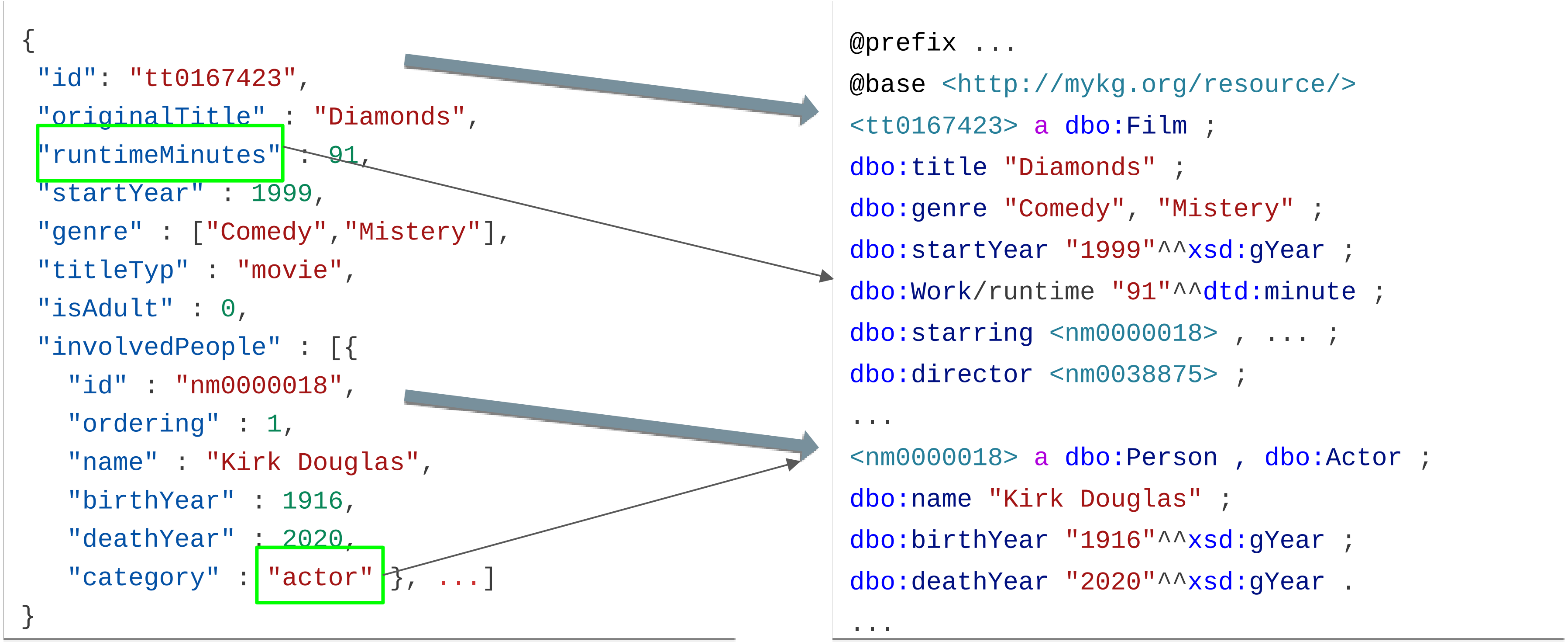
```
@prefix ...
@base <http://mykg.org/resource/>
<tt0167423> a dbo:Film ;
dbo:title "Diamonds" ;
dbo:genre "Comedy", "Mistery" ;
dbo:startYear "1999"^^xsd:gYear ;
dbo:Work/runtime "91"^^dtd:minute ;
dbo:starring <nm0000018> , ... ;
dbo:director <nm0038875> ;
...
<nm0000018> a dbo:Person , dbo:Actor ;
dbo:name "Kirk Douglas" ;
dbo:birthYear "1916"^^xsd:gYear ;
dbo:deathYear "2020"^^xsd:gYear .
...
```

Snippet of Input Data and Expected RDF Graph

```
{
 "id": "tt0167423",
 "originalTitle" : "Diamonds",
 "runtimeMinutes" : 91,
 "startYear" : 1999,
 "genre" : ["Comedy","Mistery"],
 "titleTyp" : "movie",
 "isAdult" : 0,
 "involvedPeople" : [{
   "id" : "nm0000018",
   "ordering" : 1,
   "name" : "Kirk Douglas",
   "birthYear" : 1916,
   "deathYear" : 2020,
   "category" : "actor" }, ...]
}
```

```
@prefix ...
@base <http://mykg.org/resource/>
<tt0167423> a dbo:Film ;
dbo:title "Diamonds" ;
dbo:genre "Comedy", "Mistery" ;
dbo:startYear "1999"^^xsd:gYear ;
dbo:Work/runtime "91"^^dtd:minute ;
dbo:starring <nm0000018> , ... ;
dbo:director <nm0038875> ;
...
<nm0000018> a dbo:Person , dbo:Actor ;
dbo:name "Kirk Douglas" ;
dbo:birthYear "1916"^^xsd:gYear ;
dbo:deathYear "2020"^^xsd:gYear .
...
```

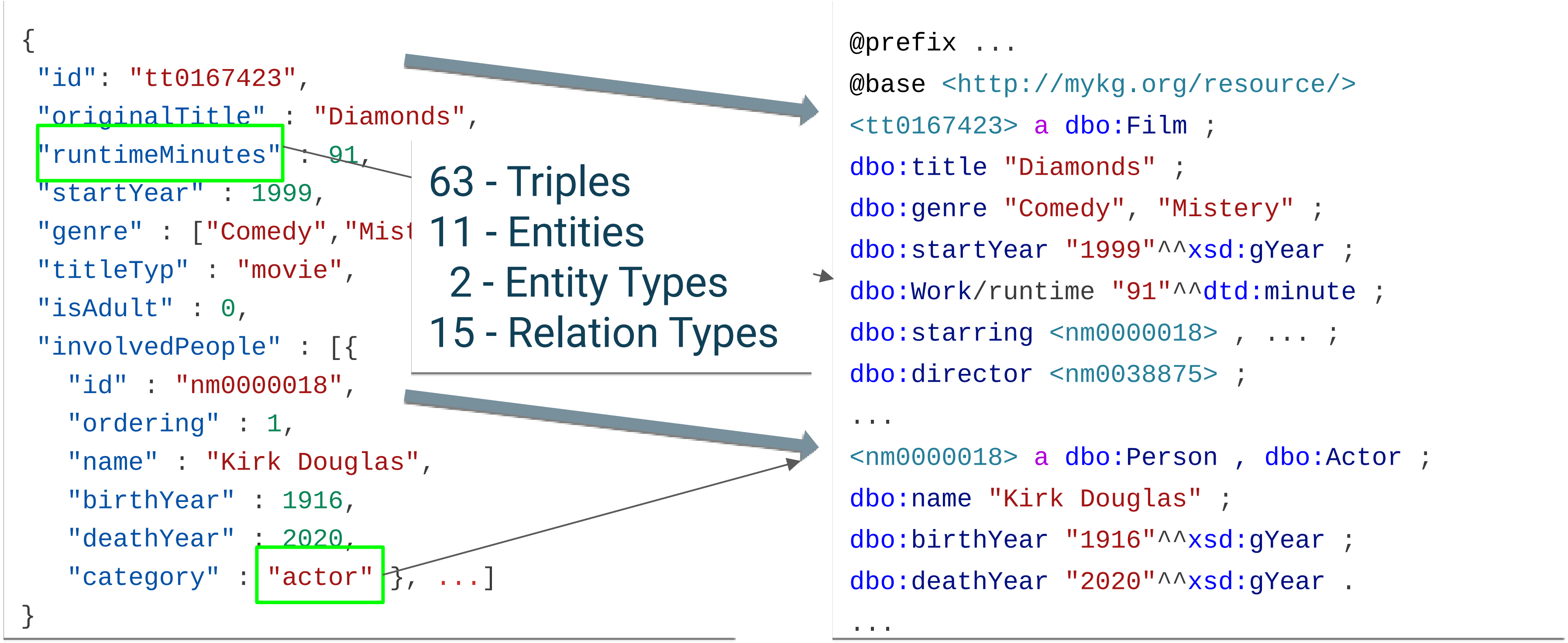
Snippet of Input Data and Expected RDF Graph



```
{
 "id": "tt0167423",
 "originalTitle" : "Diamonds",
 "runtimeMinutes" : 91,
 "startYear" : 1999,
 "genre" : ["Comedy","Mistery"],
 "titleTyp" : "movie",
 "isAdult" : 0,
 "involvedPeople" : [{
   "id" : "nm0000018",
   "ordering" : 1,
   "name" : "Kirk Douglas",
   "birthYear" : 1916,
   "deathYear" : 2020,
   "category" : "actor" }, ...]
}
```

```
@prefix ...
@base <http://mykg.org/resource/>
<tt0167423> a dbo:Film ;
dbo:title "Diamonds" ;
dbo:genre "Comedy", "Mistery" ;
dbo:startYear "1999"^^xsd:gYear ;
dbo:Work/runtime "91"^^dtd:minute ;
dbo:starring <nm0000018> , ... ;
dbo:director <nm0038875> ;
...
<nm0000018> a dbo:Person , dbo:Actor ;
dbo:name "Kirk Douglas" ;
dbo:birthYear "1916"^^xsd:gYear ;
dbo:deathYear "2020"^^xsd:gYear .
...
```

Snippet of Input Data and Expected RDF Graph



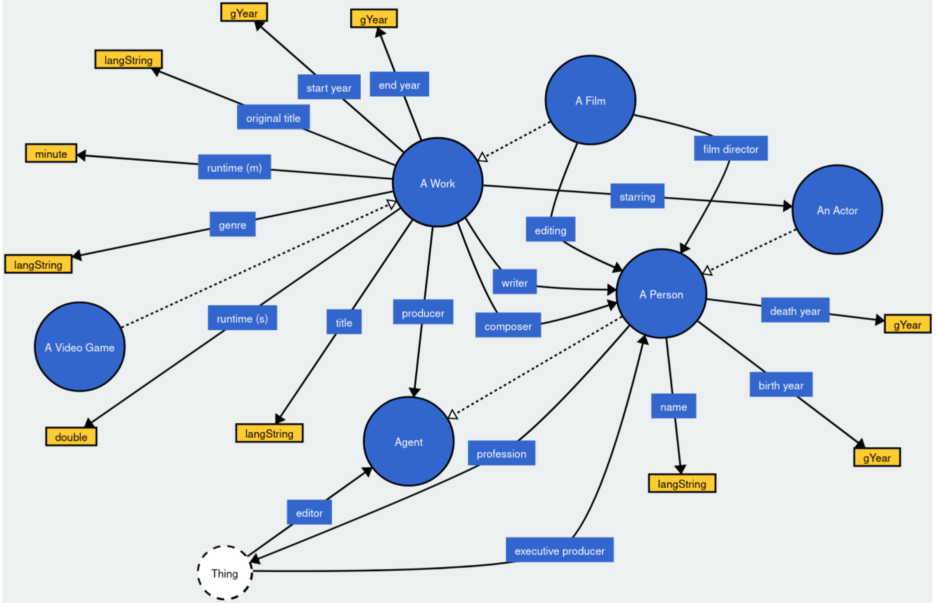
```
{
 "id": "tt0167423",
 "originalTitle" : "Diamonds",
 "runtimeMinutes" : 91,
 "startYear" : 1999,
 "genre" : ["Comedy","Mist
 "titleTyp" : "movie",
 "isAdult" : 0,
 "involvedPeople" : [{
   "id" : "nm0000018",
   "ordering" : 1,
   "name" : "Kirk Douglas",
   "birthYear" : 1916,
   "deathYear" : 2020,
   "category" : "actor" }, ...]
}
```

63 - Triples
11 - Entities
2 - Entity Types
15 - Relation Types

```
@prefix ...
@base <http://mykg.org/resource/>
<tt0167423> a dbo:Film ;
dbo:title "Diamonds" ;
dbo:genre "Comedy", "Mistery" ;
dbo:startYear "1999"^^xsd:gYear ;
dbo:Work/runtime "91"^^dtd:minute ;
dbo:starring <nm0000018> , ... ;
dbo:director <nm0038875> ;
...
<nm0000018> a dbo:Person , dbo:Actor ;
dbo:name "Kirk Douglas" ;
dbo:birthYear "1916"^^xsd:gYear ;
dbo:deathYear "2020"^^xsd:gYear .
...
```

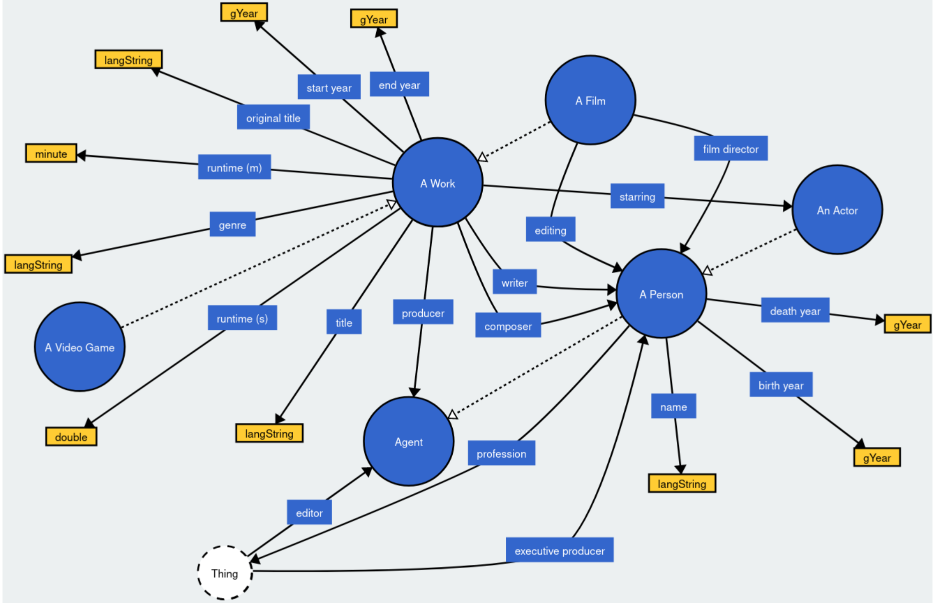
Ontology Development

- Serves the ***purpose to check whether the LLms are capable*** of using it correctly
- Selected as a ***subset of the existing DBpedia ontology*** from over 1.3K Classes, 50K Properties
- Including ***sub class, sub property, labels, and comments*** properties
- Represented in ***RDF Turtle format*** using rdfs and owl vocabulary

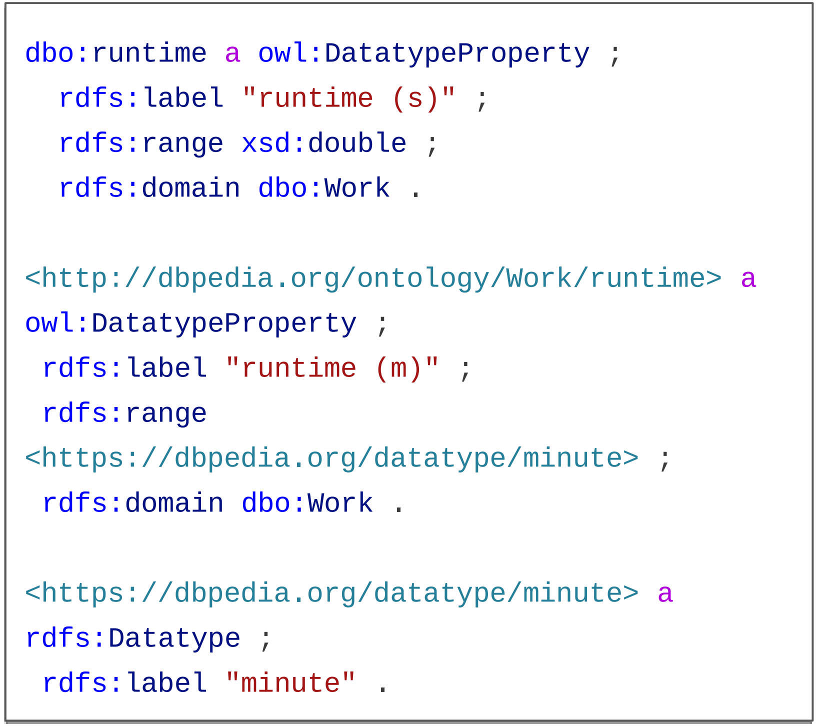


Ontology Development

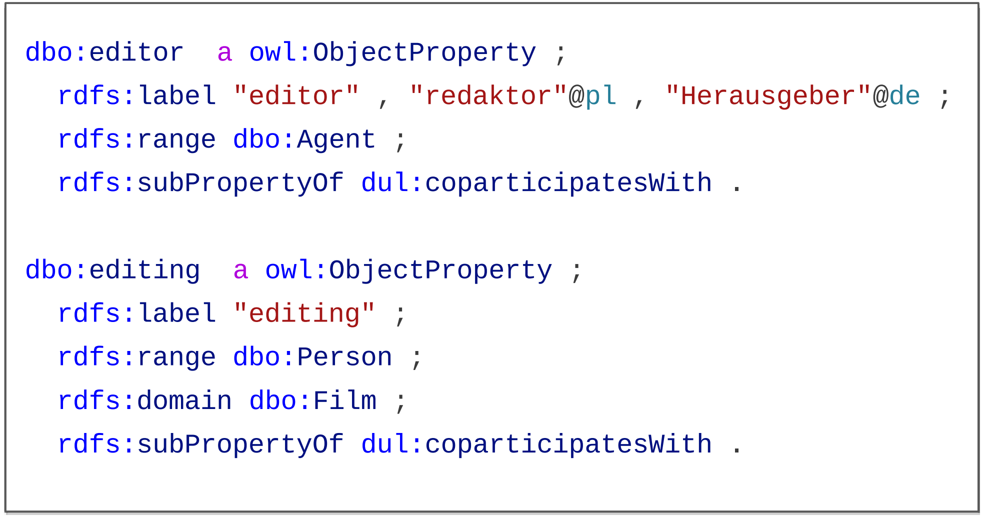
- **Entity Types:** Person > Actor, Work > Film, (VideoGame)
- **Datatype Properties:** runtime (Work/runtime), birthYear, deathYear, genre, name, originalTitle, startYear, title
- **Object Properties:** rdf:type, composer, director, editing executiveProducer, starring, writer (editor, producer, profession)
- **Not Mapped:** isAdult & ordering



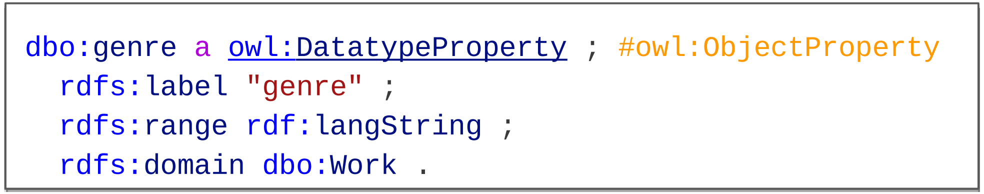
Confusable Properties



```
dbo:runtime a owl:DatatypeProperty ;
  rdfs:label "runtime (s)" ;
  rdfs:range xsd:double ;
  rdfs:domain dbo:Work .

<http://dbpedia.org/ontology/Work/runtime> a
owl:DatatypeProperty ;
 rdfs:label "runtime (m)" ;
 rdfs:range
<https://dbpedia.org/datatype/minute> ;
 rdfs:domain dbo:Work .

<https://dbpedia.org/datatype/minute> a
rdfs:Datatype ;
 rdfs:label "minute" .
```



```
dbo:editor  a owl:ObjectProperty ;
  rdfs:label "editor" , "redaktor"@pl , "Herausgeber"@de ;
  rdfs:range dbo:Agent ;
  rdfs:subPropertyOf dul:coparticipatesWith .

dbo:editing  a owl:ObjectProperty ;
  rdfs:label "editing" ;
  rdfs:range dbo:Person ;
  rdfs:domain dbo:Film ;
  rdfs:subPropertyOf dul:coparticipatesWith .
```



```
dbo:genre a owl:DatatypeProperty ; #owl:ObjectProperty
  rdfs:label "genre" ;
  rdfs:range rdf:langString ;
  rdfs:domain dbo:Work .
```

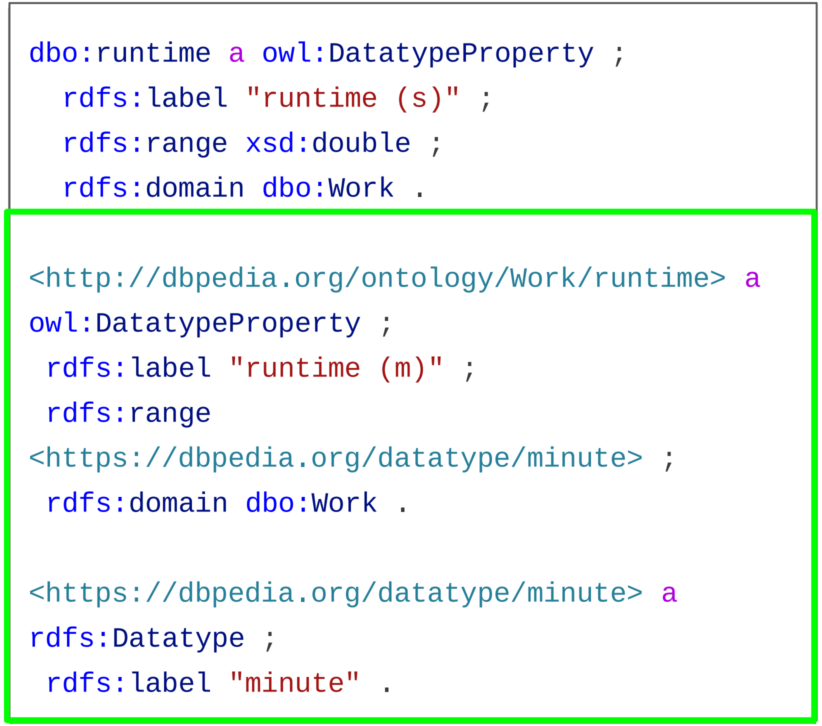
Confusable Properties

```
dbo:runtime a owl:DatatypeProperty ;
  rdfs:label "runtime (s)" ;
  rdfs:range xsd:double ;
  rdfs:domain dbo:Work .

<http://dbpedia.org/ontology/Work/runtime> a
owl:DatatypeProperty ;
 rdfs:label "runtime (m)" ;
 rdfs:range
<https://dbpedia.org/datatype/minute> ;
 rdfs:domain dbo:Work .

<https://dbpedia.org/datatype/minute> a
rdfs:Datatype ;
 rdfs:label "minute" .
```

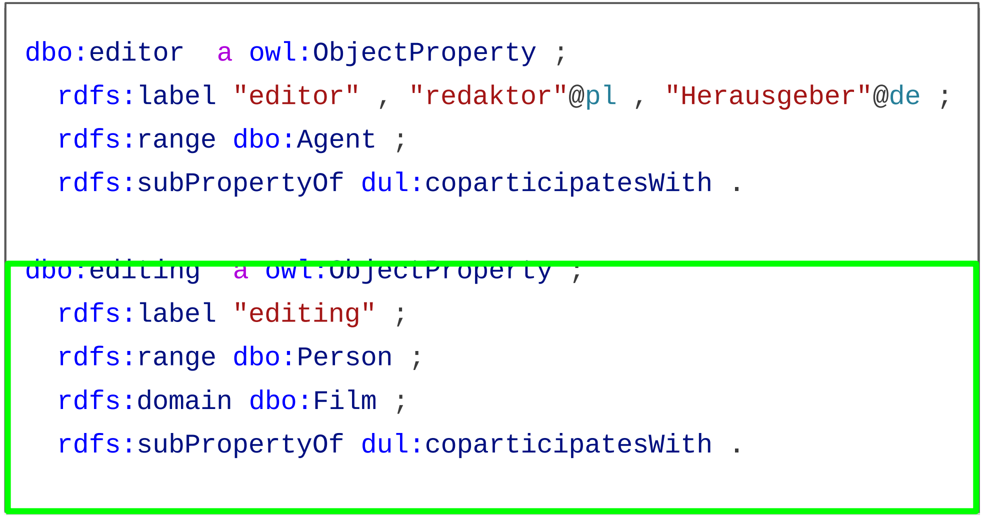
```
dbo:editor  a owl:ObjectProperty ;
  rdfs:label "editor" , "redaktor"@pl , "Herausgeber"@de ;
  rdfs:range dbo:Agent ;
  rdfs:subPropertyOf dul:coparticipatesWith .

dbo:editing  a owl:ObjectProperty ;
  rdfs:label "editing" ;
  rdfs:range dbo:Person ;
  rdfs:domain dbo:Film ;
  rdfs:subPropertyOf dul:coparticipatesWith .
```

```
dbo:genre a owl:DatatypeProperty ; #owl:ObjectProperty
  rdfs:label "genre" ;
  rdfs:range rdf:langString ;
  rdfs:domain dbo:Work .
```

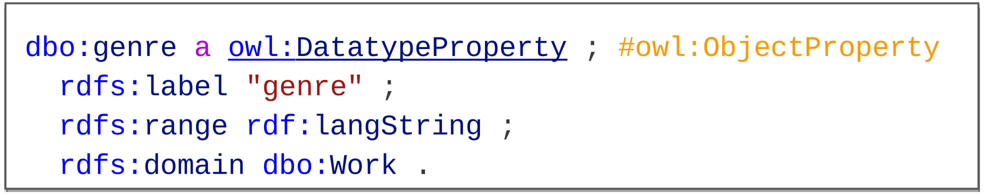
Confusable Properties



```
dbo:runtime a owl:DatatypeProperty ;
  rdfs:label "runtime (s)" ;
  rdfs:range xsd:double ;
  rdfs:domain dbo:Work .

<http://dbpedia.org/ontology/Work/runtime> a
owl:DatatypeProperty ;
 rdfs:label "runtime (m)" ;
 rdfs:range
<https://dbpedia.org/datatype/minute> ;
 rdfs:domain dbo:Work .

<https://dbpedia.org/datatype/minute> a
rdfs:Datatype ;
 rdfs:label "minute" .
```

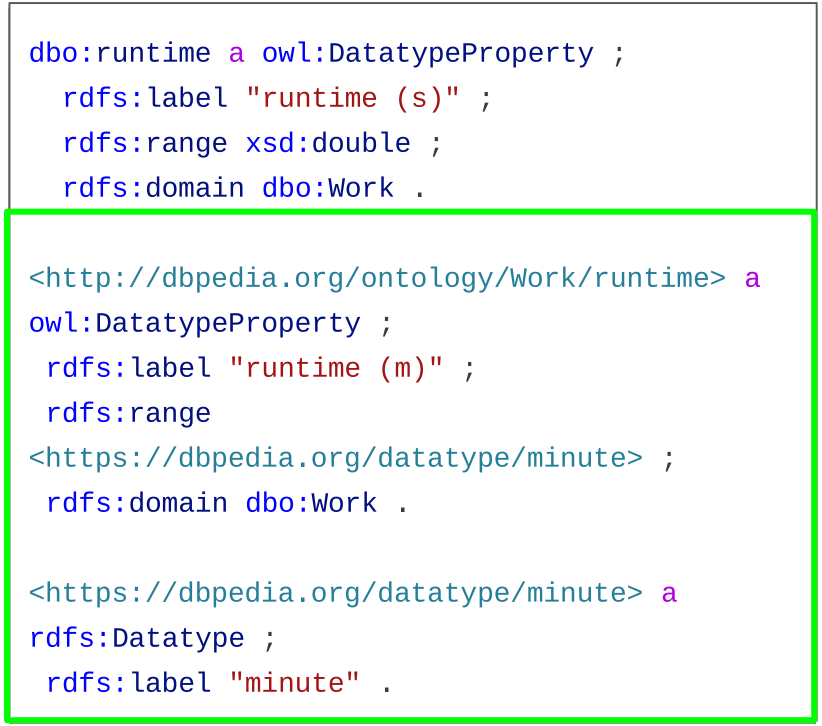


```
dbo:editor  a owl:ObjectProperty ;
  rdfs:label "editor" , "redaktor"@pl , "Herausgeber"@de ;
  rdfs:range dbo:Agent ;
  rdfs:subPropertyOf dul:coparticipatesWith .

dbo:editing  a owl:ObjectProperty ;
  rdfs:label "editing" ;
  rdfs:range dbo:Person ;
  rdfs:domain dbo:Film ;
  rdfs:subPropertyOf dul:coparticipatesWith .
```

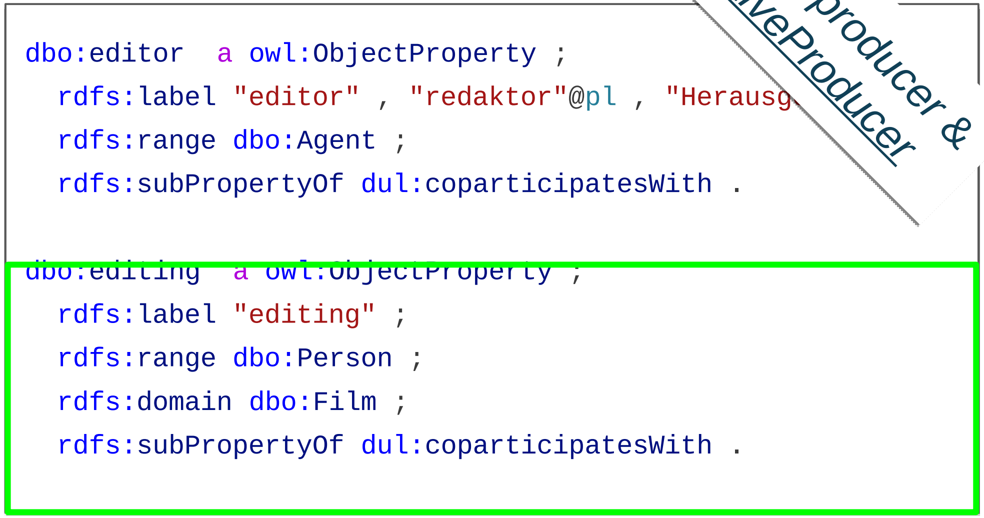


```
dbo:genre a owl:DatatypeProperty ; #owl:ObjectProperty
  rdfs:label "genre" ;
  rdfs:range rdf:langString ;
  rdfs:domain dbo:Work .
```

Confusable Properties

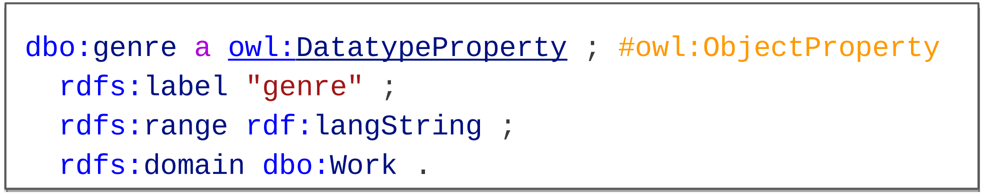


```
dbo:runtime a owl:DatatypeProperty ;
  rdfs:label "runtime (s)" ;
  rdfs:range xsd:double ;
  rdfs:domain dbo:Work .

<http://dbpedia.org/ontology/Work/runtime> a
owl:DatatypeProperty ;
 rdfs:label "runtime (m)" ;
 rdfs:range
<https://dbpedia.org/datatype/minute> ;
 rdfs:domain dbo:Work .

<https://dbpedia.org/datatype/minute> a
rdfs:Datatype ;
 rdfs:label "minute" .
```



```
dbo:editor  a owl:ObjectProperty ;
  rdfs:label "editor" , "redaktor"@pl , "Herausg
  rdfs:range dbo:Agent ;
  rdfs:subPropertyOf dul:coparticipatesWith .

dbo:editing  a owl:ObjectProperty ;
  rdfs:label "editing" ;
  rdfs:range dbo:Person ;
  rdfs:domain dbo:Film ;
  rdfs:subPropertyOf dul:coparticipatesWith .
```

Same for producer & executiveProducer



```
dbo:genre a owl:DatatypeProperty ; #owl:ObjectProperty
  rdfs:label "genre" ;
  rdfs:range rdf:langString ;
  rdfs:domain dbo:Work .
```

Prompt Engineering

- Started with a single simple prompts
- ***"Generate an RML mapping in Turtle for the given JSON to the given RDF Ontology"***
- Adapted the prompt by adding further instructions based on observed issues
- Two final prompts
  1. RML Generation Prompt
  2. RDF Repair Prompt

RML Generation Prompt

You are a helpful assistant that provides full RML mappings in RDF turtle format that aim to convert a full JSON input file source (assume filename /path/to/input.json) into RDF using the provided DBpedia movie ontology as mapping target.

You will be given a representative sample from the input source in order to derive generic information for the schema of the file. Map information as fine-grained as possible w.r.t. the target ontology, by identifying the best matches for classes, properties and only use more generic (coarse-grained) classes/properties from the target ontology when there are no better matches. Only create mappings to classes or properties defined by the given target ontology. Take the domain and range definitions of properties into account and use RML (builtin only) transformation functions to convert input according to the expected output datatype whenever necessary and possible. You shall use information about domain and ranges from the given target ontology. Make sure the mapping is syntactically and semantically correct to the RML specification or RML ontology such that it can be automatically processed. Use the http://mykg.org/resource/ namespace for creating the subject IRIs.

{ONTOLOGY TURTL}

{JSON INPUT}

RML Generation Prompt

You are a helpful assistant that provides full RML mappings in RDF turtle format that aim to convert a full JSON input file source (assume filename /path/to/input.json) into RDF using the provided DBpedia movie ontology as mapping target.

You will be given a na
of the file. Map info for
classes, properties en
there are no better ke
the domain and ra to
convert input accor on
about domain and lly
correct to the RM he
http://mykg.org/reso

{ONTOLOGY TURT

{JSON INPUT}

- Convert given JSON data with file **source located at /path/to/input.json**
- Use the provided **movie ontology as a mapping target**
- Map information with the **most specific class or property** possible
- Take the **domain and range of properties** into account
- (**Convert values** to be valid for datatypes)
- Ensure syntactic and semantic **correctness to RML specification**
- Use **'http://mykg.org/resource/'** as target namespace

RDF Repair Prompt

You are a helpful assistant that repairs broken RDF Turtle syntax, given as input by the user.

Stick with the original structure and formatting of the file as much as possible. Try to fix it with minor modifications of single character or symbols, especially do not remove any lines and triples unless there is no syntax fix possible, and also do not add information to the file, that was not stated before. Please take care that the file has proper usage of the comma, semicolon, and dot symbols in the turtle syntax: According to the W3C RDF 1.1 Turtle Terse RDF Triple Language specification the ';' symbol is used to repeat the same subject for triples that vary only in predicate and object RDF terms, only use '.' when defining a new subject in the next triple. The same applies when using ']' notation, append '.' when defining a new subject in the subsequent triple. The ',' is is used to enumerate multiple object for the same subject-predicate pair. Also take the given parsing exception or error message into account, but in some cases they might be misleading. Please respond with the full fixed RDF Turtle document, including all necessary prefix declarations.

{ERROR MSG}

{INVALID RDF}

RDF Repair Prompt

You are a helpful assistant that repairs broken RDF Turtle syntax, given as input by the user.

Stick with the original structure and formatting of the file as much as possible. Try to fix it with minor modifications
of single characte k
possible, and also s
proper usage of th L
Turtle Terse RDF T t
vary only in predica e
applies when using d
to enumerate mult r
message into acco e
document, includin

- Respond with the **full fixed RDF Turtle** document.
- **Stick with the original** structure and formatting of the original Turtle file as much as possible.
- Only apply **minor modifications** to fix the syntax.
- Take the given **parsing exception into account** when repairing
- Check proper **usage of** . , ; for separating triples, predicate-objects, and objects.

{ERROR MSG}

{INVALID RDF}

RML Mapping Requirements & Challenges

RML Mapping Requirements & Challenges

- defining correct **logical source** based on given file path

RML Mapping Requirements & Challenges

- defining correct **logical source** based on given file path
- **mapping all** JSON **attributes** where a target property (candidate) **exists in the ontology** but not mapping keys without a candidate (ontology coverage & succinctness)

RML Mapping Requirements & Challenges

- defining correct **logical source** based on given file path
- **mapping all** JSON **attributes** where a target property (candidate) **exists in the ontology** but not mapping keys without a candidate (ontology coverage & succinctness)
- **selecting most specific over generic** properties and types (e.g. Actor instead of Person) w.r.t. formalized context of ontology members

RML Mapping Requirements & Challenges

- defining correct **logical source** based on given file path
- **mapping all** JSON **attributes** where a target property (candidate) **exists in the ontology** but not mapping keys without a candidate (ontology coverage & succinctness)
- **selecting most specific over generic** properties and types (e.g. Actor instead of Person) w.r.t. formalized context of ontology members
- correct literal value representations and **datatypes**

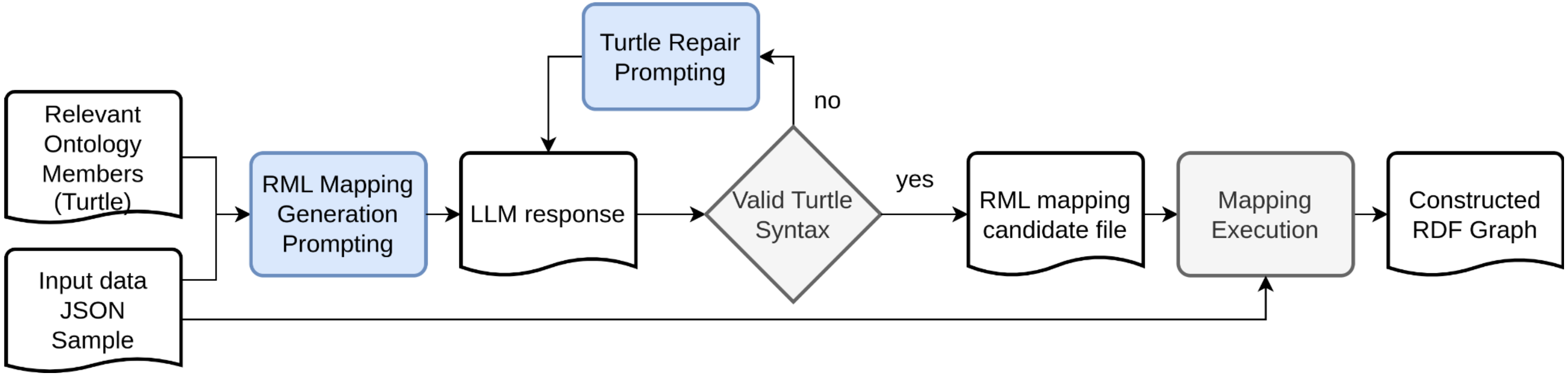
RML Mapping Requirements & Challenges

- defining correct **logical source** based on given file path
- **mapping all** JSON **attributes** where a target property (candidate) **exists in the ontology** but not mapping keys without a candidate (ontology coverage & succinctness)
- **selecting most specific over generic** properties and types (e.g. Actor instead of Person) w.r.t. formalized context of ontology members
- correct literal value representations and **datatypes**
- following a specified **pattern for entity IRIs** incorporating their IDs

RML Mapping Requirements & Challenges

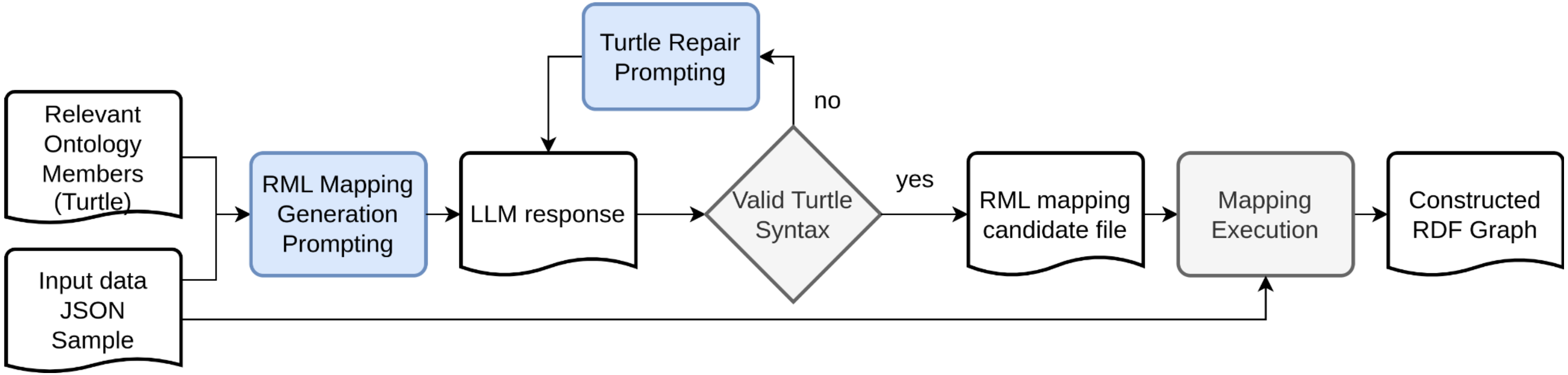
- defining correct **logical source** based on given file path
- **mapping all** JSON **attributes** where a target property (candidate) **exists in the ontology** but not mapping keys without a candidate (ontology coverage & succinctness)
- **selecting most specific over generic** properties and types (e.g. Actor instead of Person) w.r.t. formalized context of ontology members
- correct literal value representations and **datatypes**
- following a specified **pattern for entity IRIs** incorporating their IDs
- (usage of RML-Mapper **built-in functions only**)

Evaluation Setup



Evaluation Setup

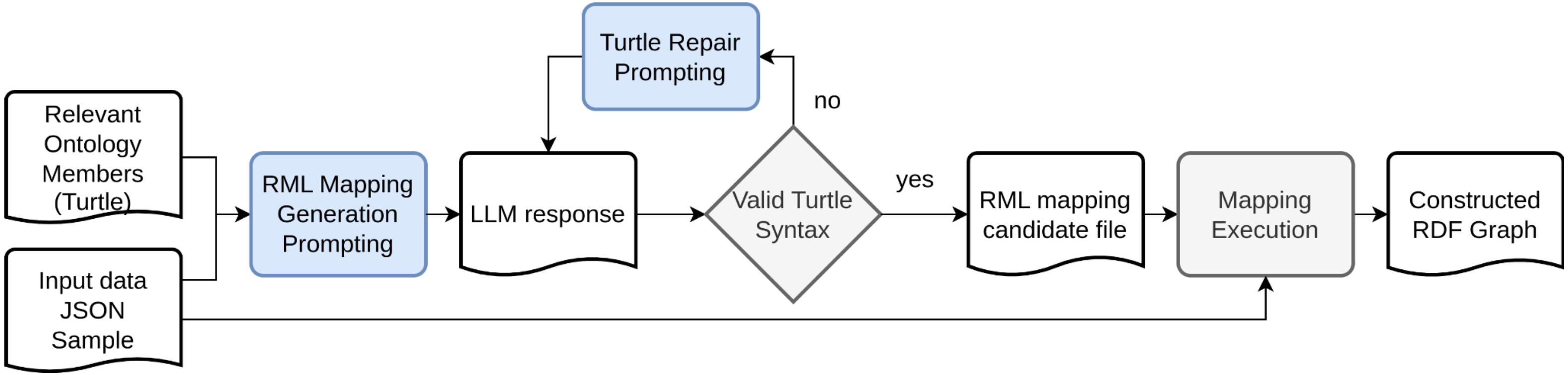
40 runs for each Model (Claude 2.1 / 3.0, GPT 3.5 / 4 Turbo, Gemini Pro)



Evaluation Setup

40 runs for each Model (Claude 2.1 / 3.0, GPT 3.5 / 4 Turbo, Gemini Pro)

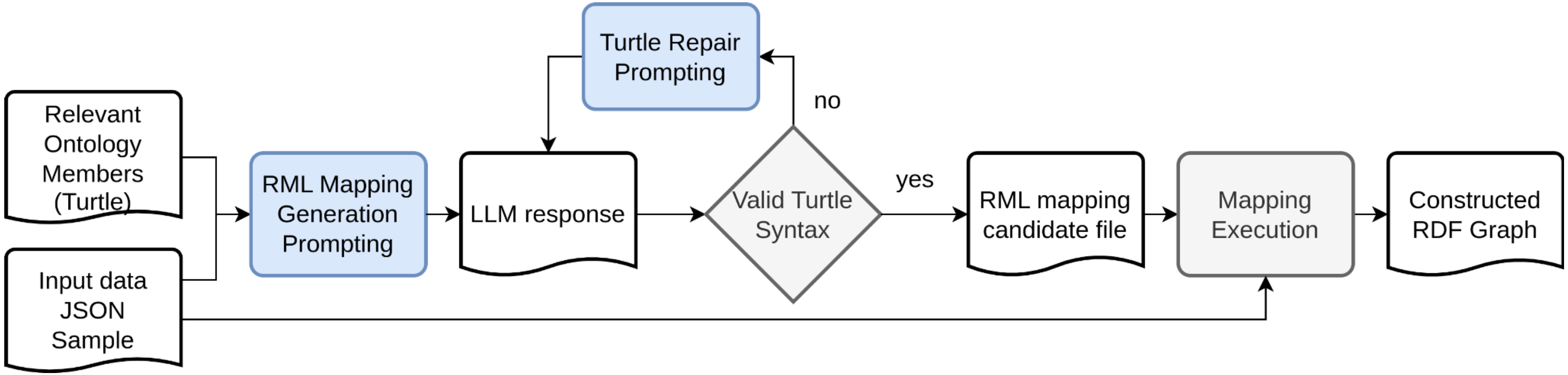
1. For each run, we **check the output for RDF syntax errors**. if invalid **up to two consecutive repair attempts.**



Evaluation Setup

40 runs for each Model (Claude 2.1 / 3.0, GPT 3.5 / 4 Turbo, Gemini Pro)

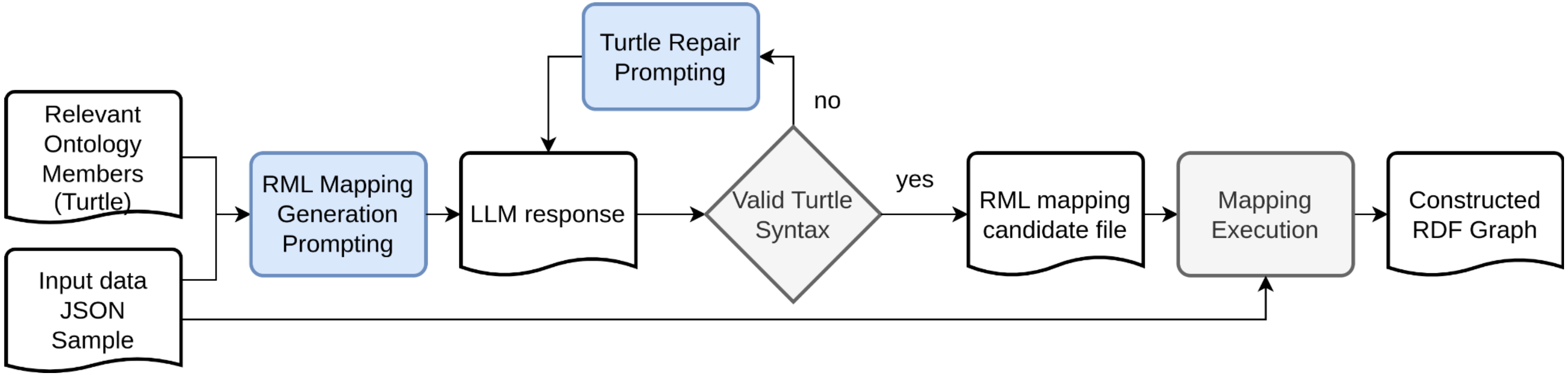
1. For each run, we **check the output for RDF syntax errors**. if invalid **up to two consecutive repair attempts.**
2. If successfully (repaired), we **evaluate the generation of triples.**



Evaluation Setup

40 runs for each Model (Claude 2.1 / 3.0, GPT 3.5 / 4 Turbo, Gemini Pro)

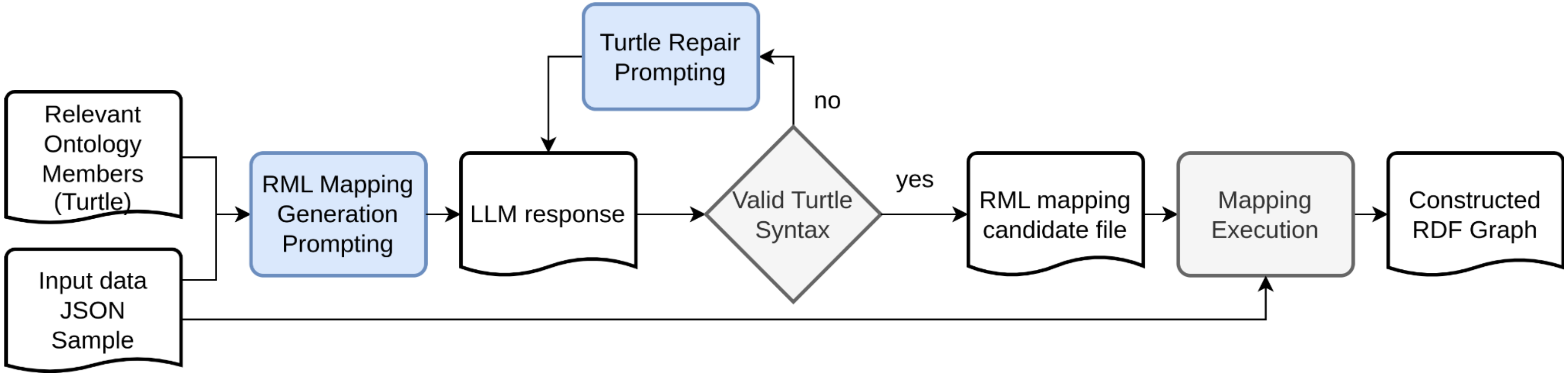
1. For each run, we **check the output for RDF syntax errors**. if invalid **up to two consecutive repair attempts.**
2. If successfully (repaired), we **evaluate the generation of triples.**
3. Then verify the **correctness of these triples,**



Evaluation Setup

40 runs for each Model (Claude 2.1 / 3.0, GPT 3.5 / 4 Turbo, Gemini Pro)

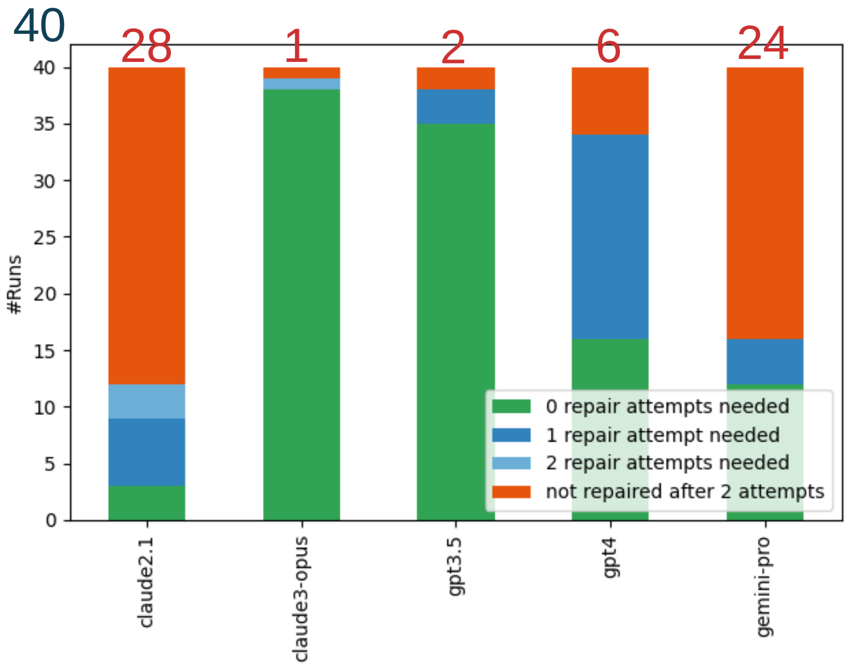
1. For each run, we **check the output for RDF syntax errors**. if invalid **up to two consecutive repair attempts.**
2. If successfully (repaired), we **evaluate the generation of triples.**
3. Then verify the **correctness of these triples,**
4. Finally, we assess if **correctly mapped to the target ontology**



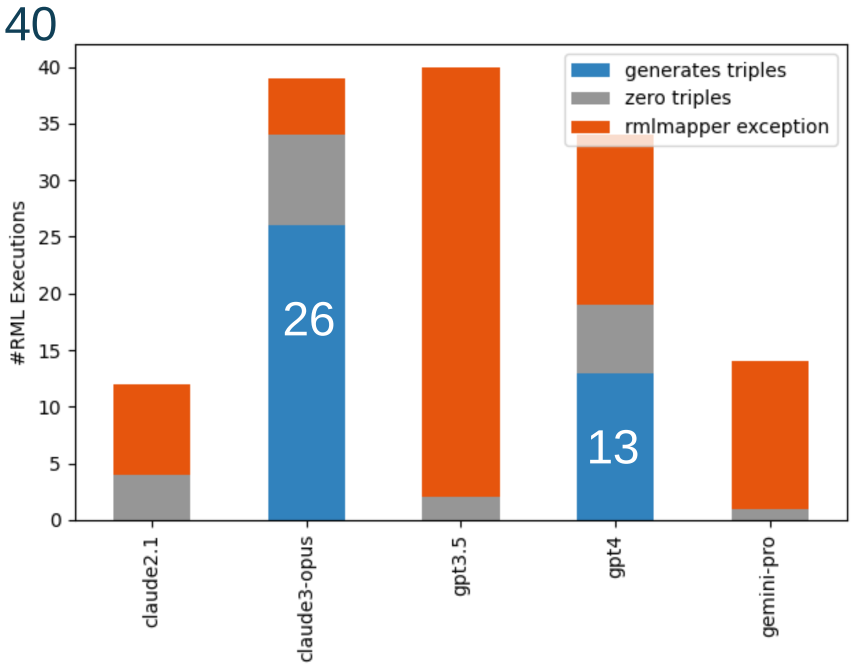
Evaluation LLM Response Validity

```
rr:template
"...mykg.org/…/{$.involvedPeople[
?(@.category=='editor')].id}"
```

RDF Turtle Syntax Validity



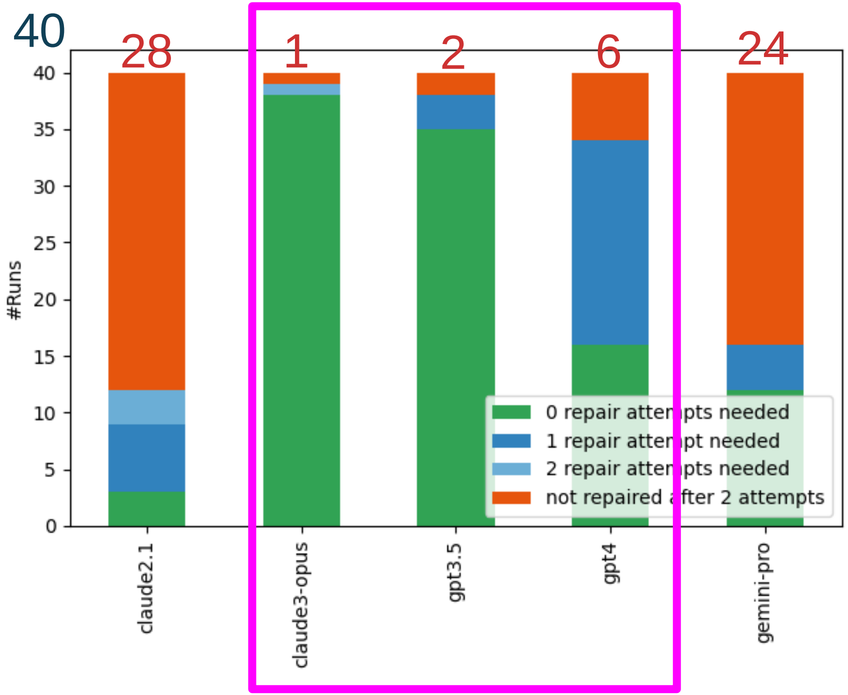
Mapping Soundness (how valid is declaration)



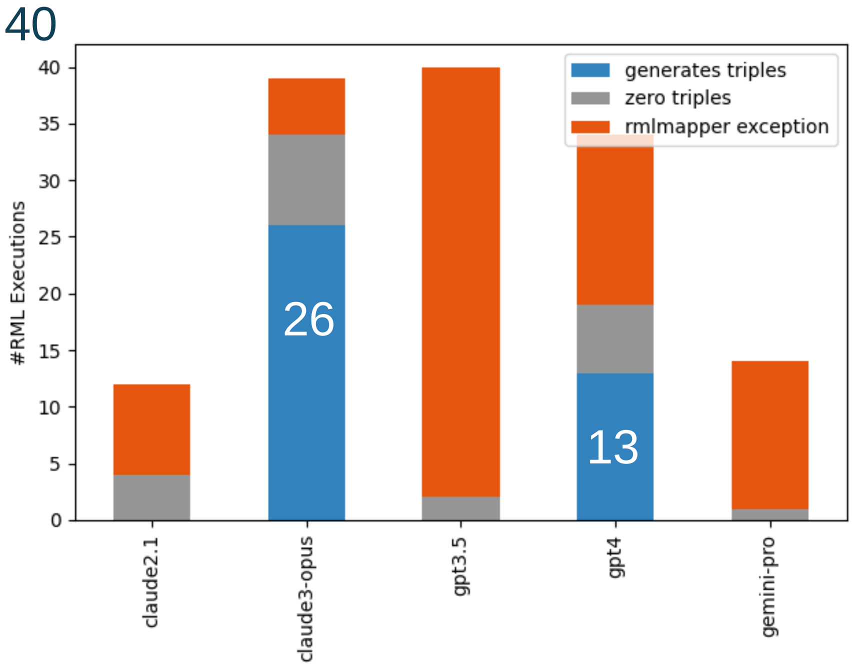
Evaluation LLM Response Validity

```
rr:template
"...mykg.org/…/{$.involvedPeople[
?(@.category=='editor')].id}"
```

RDF Turtle Syntax Validity



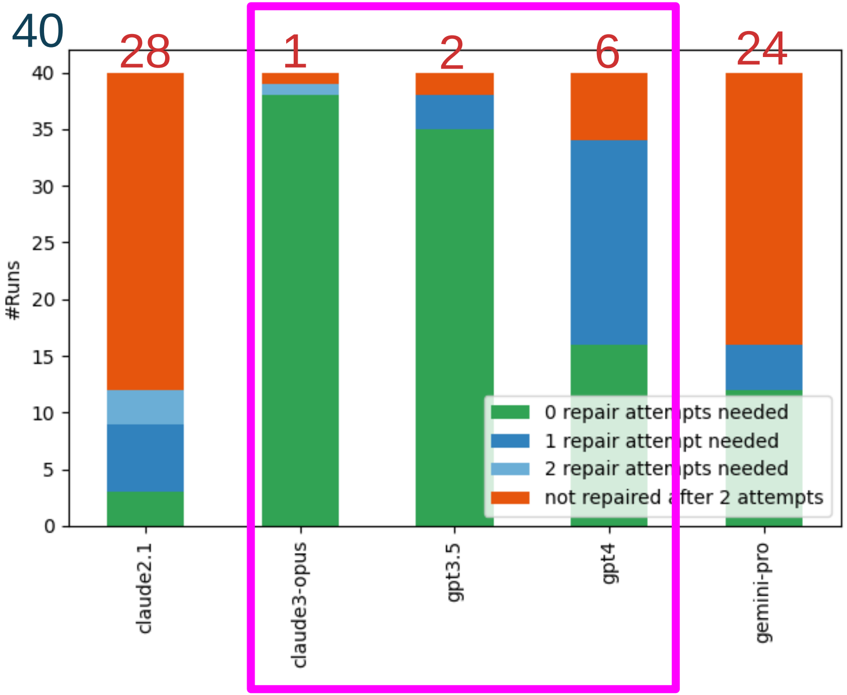
Mapping Soundness (how valid is declaration)



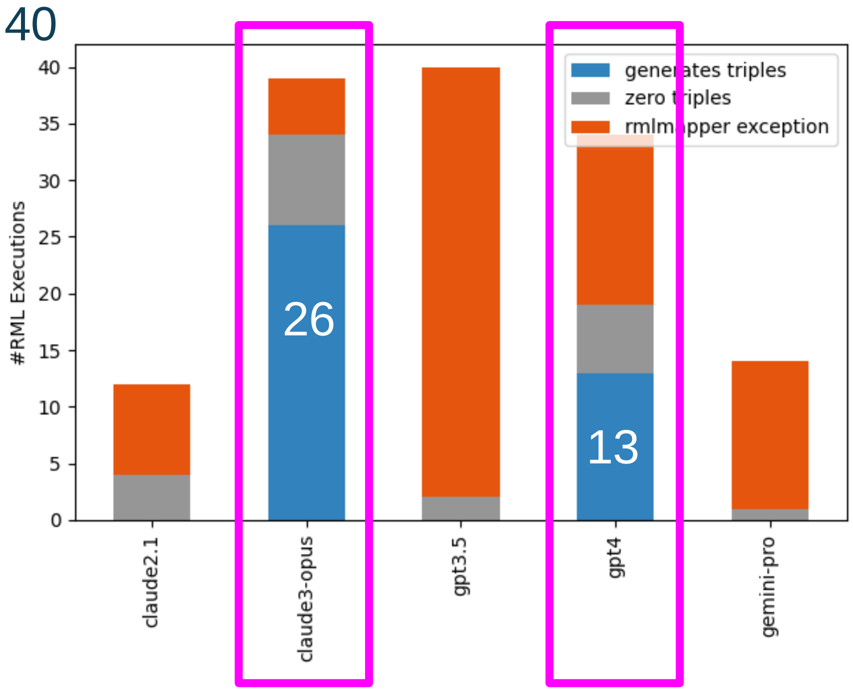
Evaluation LLM Response Validity

```
rr:template
"...mykg.org/…/{$.involvedPeople[
?(@.category=='editor')].id}"
```

RDF Turtle Syntax Validity



Mapping Soundness (how valid is declaration)



Simple query-based evaluation

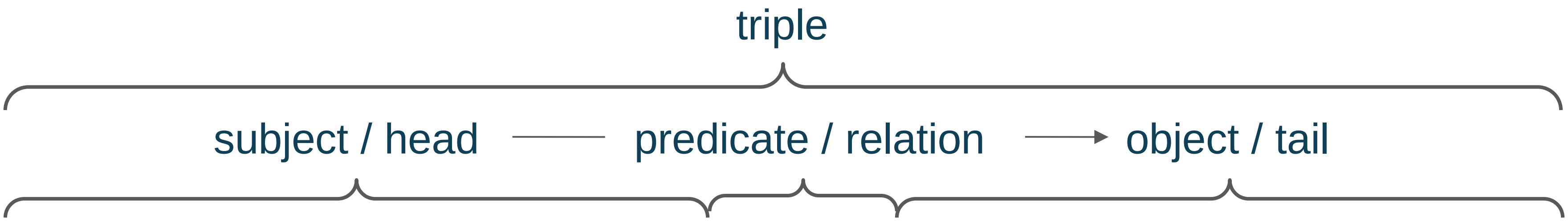
Simple query-based evaluation

	Claude 3	Gpt 4
Mappings with triples	**26 (100%)**	**13 (100%)**
All **people** entities have IRI containing **correct ID field**	**21 (80%)**	9 (70%)
All **people** IRIs are **typed**	**20 (77%)**	7 (54%)
All **actors** entities have IRI containing **correct ID field**	**21 (80%)**	9 (70%)
All **actor** IDs are **typed**	**11 (42%)**	0 (0%)
Full predicate coverage	**4 (15%)**	0 (0%)
Only ontology mapped	**20 (77%)**	2 (15%)
isAdult or ordering **not mapped**	**26 (100%)**	**13 (100%)**
Usage of **any / custom function**	**0/0 (-)**	3/3 (-)

⇒ At first glance, Claude 3 outperforms GPT 4

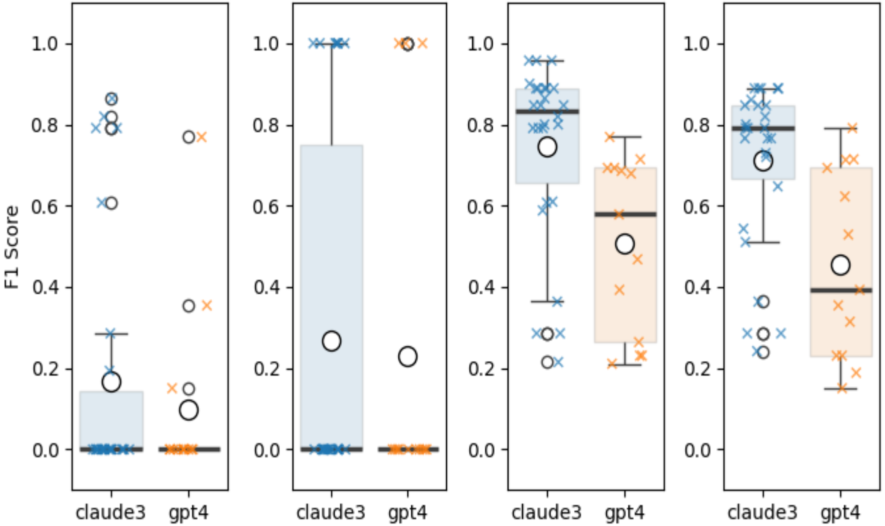
Triple Exact Match Comparison

- Very strict set of scores that report 4 graph identity measures
  - **triples, subject IRIs, predicate IRIs, object IRIs/Literals**
- F1 scores are calculated based on generated, correct, and reference sets



```
<http://mykg.org/resource/nm000018> rdf:type <http://dbpedia.org/ontology/Actor> .
<http://mykg.org/resource/nm000018> rdfs:label “Kirk Douglas”@en .
<http://mykg.org/resource/nm000018> dbo:birthYear “1900”^^xsd:gYear .
```

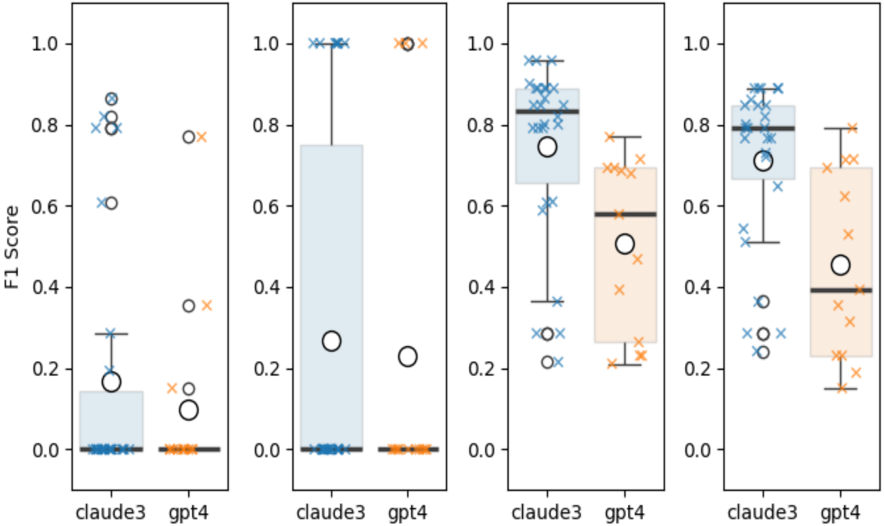
Triple Exact Match Comparison



Triple Subject Predicate Object

Triple Exact Match Comparison

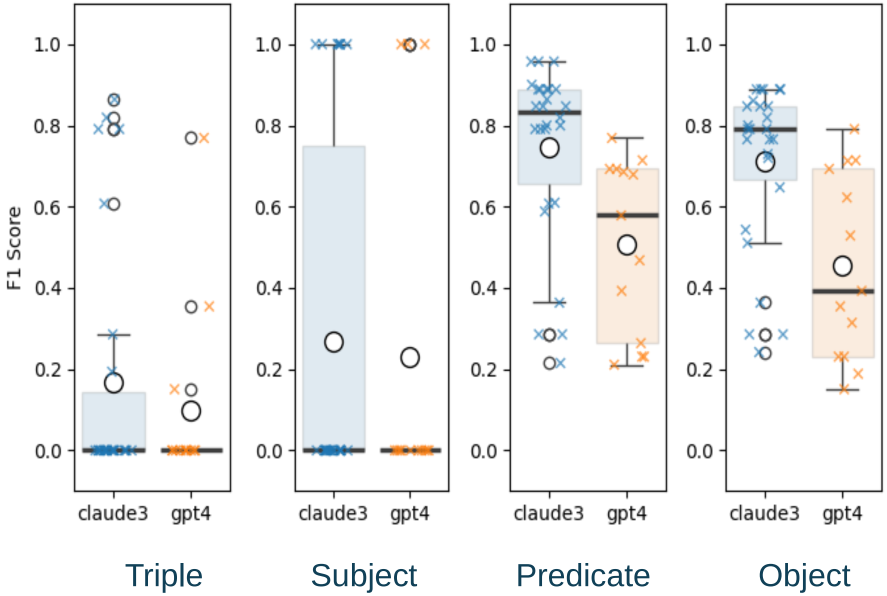
- Exact Triple match is below 0.2



Triple Subject Predicate Object

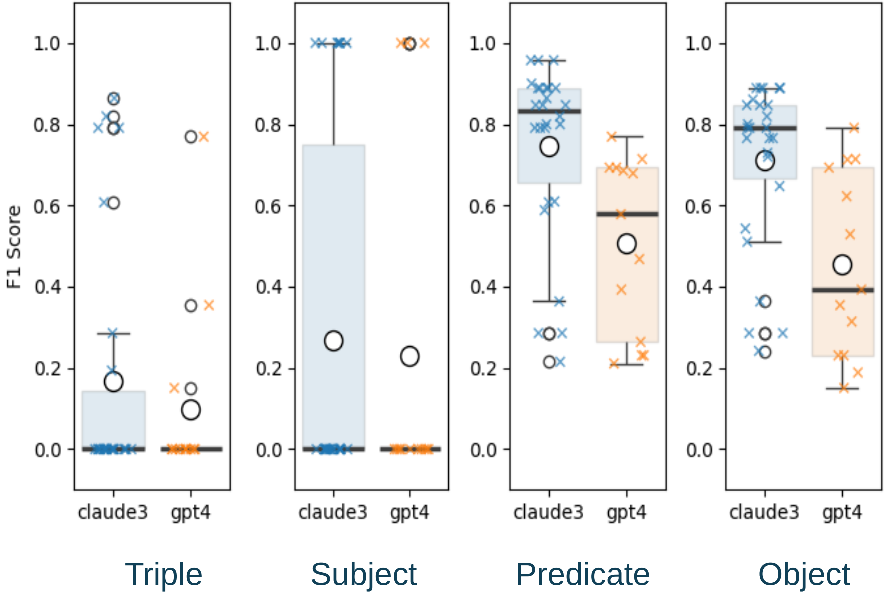
Triple Exact Match Comparison

- Exact Triple match is below 0.2
- Most mismatches for Subject IRIs



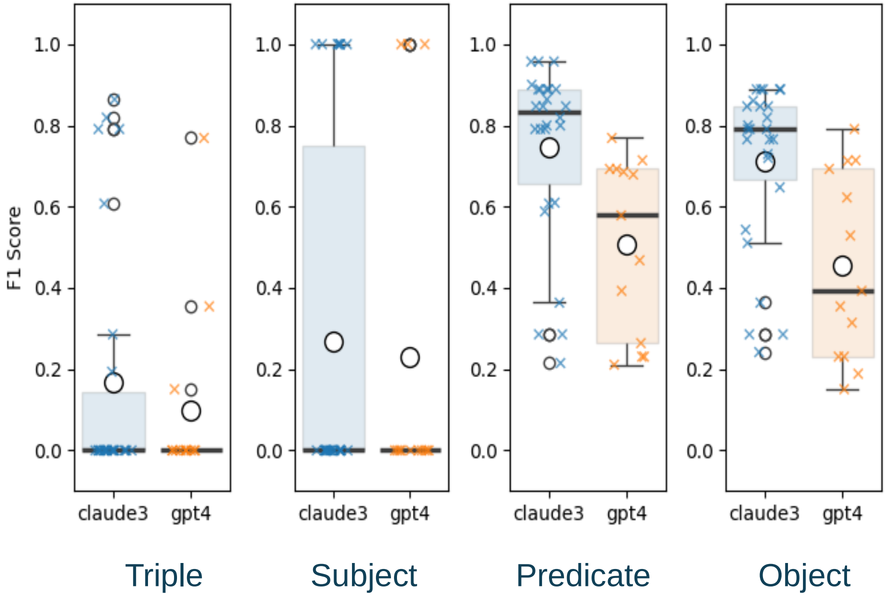
Triple Exact Match Comparison

- Exact Triple match is below 0.2
- Most mismatches for Subject IRIs
- Claude 3 reaches F1 mean of arrouch 0.75 for predicate and object match



Triple Exact Match Comparison

- Exact Triple match is below 0.2
- Most mismatches for Subject IRIs
- Claude 3 reaches F1 mean of arrouch 0.75 for predicate and object match
- Claude 3 again better than GPT 4

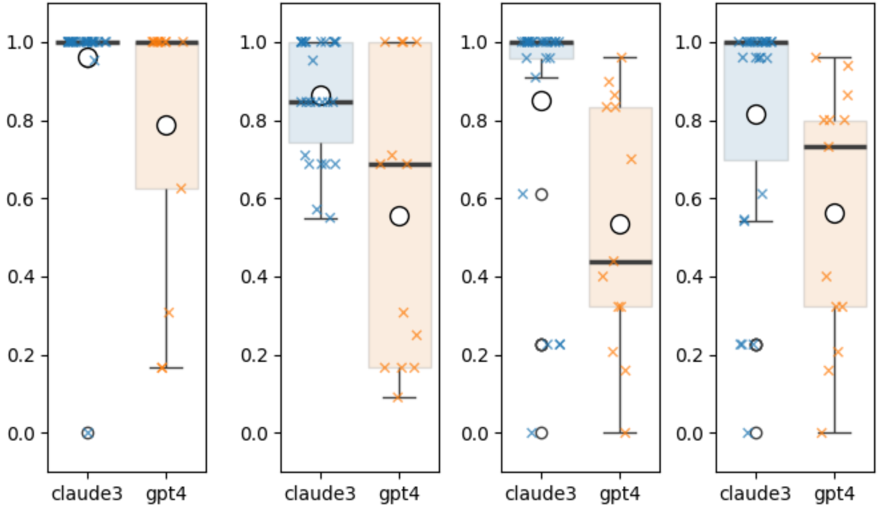


Relaxed Scores

<mykg.org/id/t1000>
<mykg.org/*t1000*>

1 Film
10 Persons
4 Actors

“1916”^^xsd:gYear .
“1916”^^xsd:date .
“2020”^^xsd:gYear .



E-ID E-Type Value Datatype

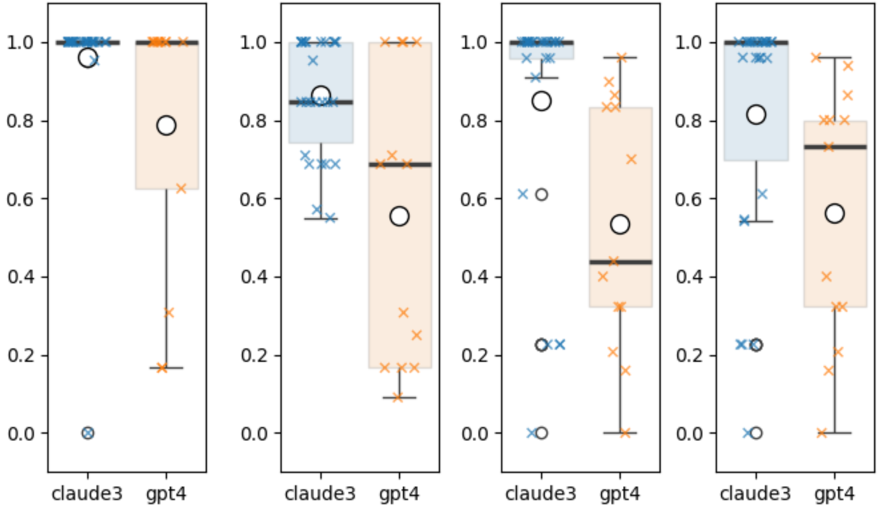
Relaxed Scores

- Much better scores than Strict Exact Measures

<mykg.org/id/t1000>
<mykg.org/*t1000*>

1 Film
10 Persons
4 Actors

“1916”^^xsd:gYear .
“1916”^^xsd:date .
“2020”^^xsd:gYear .



E-ID E-Type Value Datatype

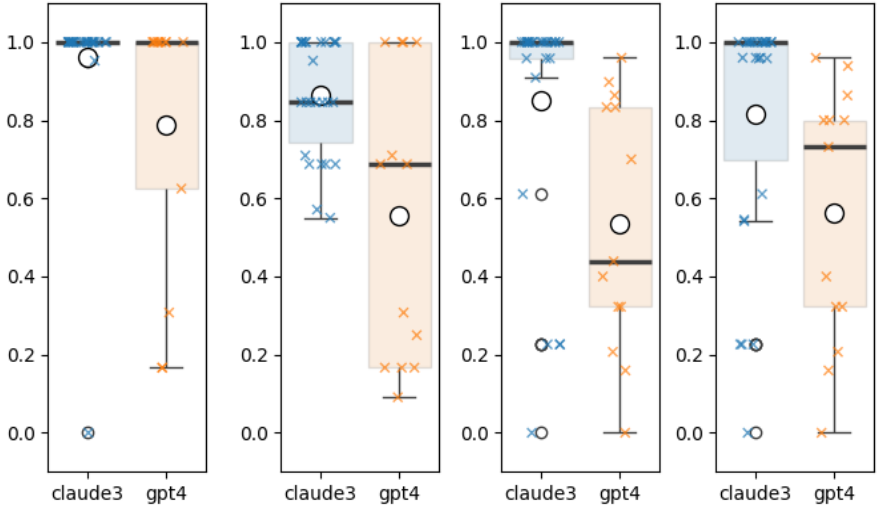
Relaxed Scores

- Much better scores than Strict Exact Measures
- E-ID score (relaxed Subject IRI) is almost 1 for Claude 3

<mykg.org/id/t1000>
<mykg.org/*t1000*>

1 Film
10 Persons
4 Actors

“1916”^^xsd:gYear .
“1916”^^xsd:date .
“2020”^^xsd:gYear .



E-ID E-Type Value Datatype

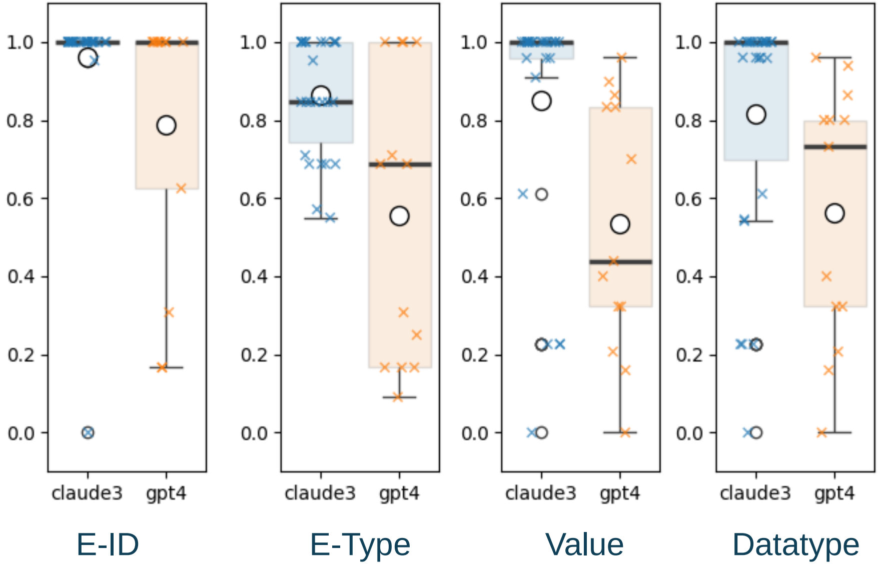
Relaxed Scores

- Much better scores than Strict Exact Measures
- E-ID score (relaxed Subject IRI) is almost 1 for Claude 3
- GPT 4 has more trouble with Literal mappings than Claude 3 (for both parts, the value and datatype)

<mykg.org/id/t1000>
<mykg.org/*t1000*>

1 Film
10 Persons
4 Actors

“1916”^^xsd:gYear .
“1916”^^xsd:date .
“2020”^^xsd:gYear .



Property Mappings Insights

Property Mappings Insights

For each predicate/property and per generated RML declaration / document

Property Mappings Insights

For each predicate/property and per generated RML declaration / document

- **“property is used”** number of triples containing this property

Property Mappings Insights

For each predicate/property and per generated RML declaration / document

- **“property is used”** number of triples containing this property
- **“property outdegress is OK”** is mapped the expected number of times

Property Mappings Insights

For each predicate/property and per generated RML declaration / document

- **“property is used”** number of triples containing this property
- **“property outdegress is OK”** is mapped the expected number of times
- **“subject (fuzzy) is OK”** mapped for the right (subject entity) relation

Property Mappings Insights

For each predicate/property and per generated RML declaration / document

- **“property is used”** number of triples containing this property
- **“property outdegress is OK”** is mapped the expected number of times
- **“subject (fuzzy) is OK”** mapped for the right (subject entity) relation
- **“object is IRI”** is mapped as a object property (points to another entity)

Property Mappings Insights

For each predicate/property and per generated RML declaration / document

- **“property is used”** number of triples containing this property
- **“property outdegress is OK”** is mapped the expected number of times
- **“subject (fuzzy) is OK”** mapped for the right (subject entity) relation
- **“object is IRI”** is mapped as a object property (points to another entity)
- **“object is Literal”** is mapped as a datatype property (points to a literal)

Property Mappings Insights

For each predicate/property and per generated RML declaration / document

- **“property is used”** number of triples containing this property
- **“property outdegress is OK”** is mapped the expected number of times
- **“subject (fuzzy) is OK”** mapped for the right (subject entity) relation
- **“object is IRI”** is mapped as a object property (points to another entity)
- **“object is Literal”** is mapped as a datatype property (points to a literal)
- **“object Datatype is OK”** literal has the correct datatype

Object Property Mappings

```
rr:template
"...mykg.org/…/{$.involvedPeople[
?(@.category=='editor')].id}"
```

Claude 3

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**26**	6	6	5	0	7	6
p fecq. OK	7	5	5	5	0	4	5
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**26**	6	6	5	0	7	6

GPT 4

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**13**	1	3	0	0	2	1
p fecq. OK	0	0	1	0	0	0	0
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**13**	0	2	0	0	1	0

Object Property Mappings

- Both models fail to generate correct mapping rules for all *job function* object properties

```
rr:template
"...mykg.org/…/{$.involvedPeople[
?(@.category=='editor')].id}"
```

Claude 3

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**26**	6	6	5	0	7	6
p fecq. OK	7	5	5	5	0	4	5
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**26**	6	6	5	0	7	6

GPT 4

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**13**	1	3	0	0	2	1
p fecq. OK	0	0	1	0	0	0	0
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**13**	0	2	0	0	1	0

Object Property Mappings

```
rr:template
"...mykg.org/…/{$.involvedPeople[
?(@.category=='editor')].id}"
```

- Both models fail to generate correct mapping rules for all *job function* object properties
- Claude-3 and GPT-4 do not map the hard property *executiveProducer*

Claude 3

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**26**	6	6	5	0	7	6
p fecq. OK	7	5	5	5	0	4	5
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**26**	6	6	5	0	7	6

GPT 4

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**13**	1	3	0	0	2	1
p fecq. OK	0	0	1	0	0	0	0
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**13**	0	2	0	0	1	0

InfAI® Institute for Applied Informatics

UNIVERSITÄT LEIPZIG

ScaDS.AI DRESDEN LEIPZIG

Object Property Mappings

```
rr:template
"...mykg.org/…/{$.involvedPeople[
?(@.category=='editor')].id}"
```

- Both models fail to generate correct mapping rules for all *job function* object properties
- Claude-3 and GPT-4 do not map the hard property *executiveProducer*
- Claude-3 RML mappings use the expected target property *editing* five times, but incorrectly

Claude 3

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**26**	6	6	5	0	7	6
p fecq. OK	7	5	5	5	0	4	5
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**26**	6	6	5	0	7	6

GPT 4

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**13**	1	3	0	0	2	1
p fecq. OK	0	0	1	0	0	0	0
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**13**	0	2	0	0	1	0

Object Property Mappings

```
rr:template
"...mykg.org/…/{$.involvedPeople[
?(@.category=='editor')].id}"
```

- Both models fail to generate correct mapping rules for all *job function* object properties
- Claude-3 and GPT-4 do not map the hard property *executiveProducer*
- Claude-3 RML mappings use the expected target property *editing* five times, but incorrectly
- GPT-4's mapping results do not contain a single triple using the property *editing*

Claude 3

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**26**	6	6	5	0	7	6
p fecq. OK	7	5	5	5	0	4	5
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**26**	6	6	5	0	7	6

GPT 4

	rdf:ype	composer	director	editing	executiveProducer	starring	writer
p is used	**13**	1	3	0	0	2	1
p fecq. OK	0	0	1	0	0	0	0
o fuzzy OK	-	0	0	0	0	0	0
o is Object	**13**	0	2	0	0	1	0

Datatype Property Mapping

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	24	25	25	22	23	25	25
p fecq. OK	24	22	22	22	20	25	25
o is Object	0	0	0	1	1	1	0
p is Literal	24	25	25	21	22	24	25
datatype OK	19	21	21	18	20	21	21

Claude 3

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	1	8	8	11	9	10	11
p fecq. OK	1	7	7	11	8	10	11
o is Object	0	0	0	1	0	0	0
p is Literal	1	8	8	10	9	10	11
datatype OK	0	8	8	10	8	9	10

GPT 4

Datatype Property Mapping

- Genre property used as object property once in each model result:

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	24	25	25	22	23	25	25
p fecq. OK	24	22	22	22	20	25	25
o is Object	0	0	0	1	1	1	0
p is Literal	24	25	25	21	22	24	25
datatype OK	19	21	21	18	20	21	21

Claude 3

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	1	8	8	11	9	10	11
p fecq. OK	1	7	7	11	8	10	11
o is Object	0	0	0	1	0	0	0
p is Literal	1	8	8	10	9	10	11
datatype OK	0	8	8	10	8	9	10

GPT 4

Datatype Property Mapping

- Genre property used as object property once in each model result:
  - Contrasts with changes made in our ontology

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	24	25	25	22	23	25	25
p fecq. OK	24	22	22	22	20	25	25
o is Object	0	0	0	1	1	1	0
p is Literal	24	25	25	21	22	24	25
datatype OK	19	21	21	18	20	21	21

Claude 3

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	1	8	8	11	9	10	11
p fecq. OK	1	7	7	11	8	10	11
o is Object	0	0	0	1	0	0	0
p is Literal	1	8	8	10	9	10	11
datatype OK	0	8	8	10	8	9	10

GPT 4

Datatype Property Mapping

- Genre property used as object property once in each model result:
  - Contrasts with changes made in our ontology
  - Differs from the original definition in the DBpedia ontology (as Object Property)

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	24	25	25	22	23	25	25
p fecq. OK	24	22	22	22	20	25	25
o is Object	0	0	0	1	1	1	0
p is Literal	24	25	25	21	22	24	25
datatype OK	19	21	21	18	20	21	21

Claude 3

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	1	8	8	11	9	10	11
p fecq. OK	1	7	7	11	8	10	11
o is Object	0	0	0	1	0	0	0
p is Literal	1	8	8	10	9	10	11
datatype OK	0	8	8	10	8	9	10

GPT 4

Datatype Property Mapping

- Genre property used as object property once in each model result:
  - Contrasts with changes made in our ontology
  - Differs from the original definition in the DBpedia ontology (as Object Property)
- GPT4 only uses Work/runtime once, but incorrectly

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	24	25	25	22	23	25	25
p fecq. OK	24	22	22	22	20	25	25
o is Object	0	0	0	1	1	1	0
p is Literal	24	25	25	21	22	24	25
datatype OK	19	21	21	18	20	21	21

Claude 3

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	1	8	8	11	9	10	11
p fecq. OK	1	7	7	11	8	10	11
o is Object	0	0	0	1	0	0	0
p is Literal	1	8	8	10	9	10	11
datatype OK	0	8	8	10	8	9	10

GPT 4

Datatype Property Mapping

- Genre property used as object property once in each model result:
  - Contrasts with changes made in our ontology
  - Differs from the original definition in the DBpedia ontology (as Object Property)
- GPT4 only uses Work/runtime once, but incorrectly
- Otherwise good mapping quality

Claude 3

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	24	25	25	22	23	25	25
p fecq. OK	24	22	22	22	20	25	25
o is Object	0	0	0	1	1	1	0
p is Literal	24	25	25	21	22	24	25
datatype OK	19	21	21	18	20	21	21

GPT 4

	Work/runtime	birthYear	deathYear	genre	name	originalTitle	startYear
p is used	1	8	8	11	9	10	11
p fecq. OK	1	7	7	11	8	10	11
o is Object	0	0	0	1	0	0	0
p is Literal	1	8	8	10	9	10	11
datatype OK	0	8	8	10	8	9	10

Incorrect Property Usage

	editor	producer	runtime
p is used	1	6	0
p fecq. OK	25	20	26
o is Object	1	6	0
o is Literal	0	0	0
val+type OK	26	26	26

Claude 3

	editor	producer	runtime
p is used	1	1	10
p fecq. OK	12	12	3
o is Object	0	0	0
o is Literal	1	1	10
val+type OK	13	13	13

GPT 4

Incorrect Property Usage

- A mapping for the (wrong) property editor was generated once by both models

Claude 3

	editor	producer	runtime
p is used	1	6	0
p fecq. OK	25	20	26
o is Object	1	6	0
o is Literal	0	0	0
val+type OK	26	26	26

GPT 4

	editor	producer	runtime
p is used	1	1	10
p fecq. OK	12	12	3
o is Object	0	0	0
o is Literal	1	1	10
val+type OK	13	13	13

Incorrect Property Usage

- A mapping for the (wrong) property editor was generated once by both models
- The (wrong) producer property was mapped

	editor	producer	runtime
p is used	1	6	0
p fecq. OK	25	20	26
o is Object	1	6	0
o is Literal	0	0	0
val+type OK	26	26	26

Claude 3

p is used	1	1	10
p fecq. OK	12	12	3
o is Object	0	0	0
o is Literal	1	1	10
val+type OK	13	13	13

GPT 4

Incorrect Property Usage

- A mapping for the (wrong) property editor was generated once by both models
- The (wrong) producer property was mapped
  - six times by Claude-3.

	editor	producer	runtime
p is used	1	6	0
p fecq. OK	25	20	26
o is Object	1	6	0
o is Literal	0	0	0
val+type OK	26	26	26

Claude 3

	editor	producer	runtime
p is used	1	1	10
p fecq. OK	12	12	3
o is Object	0	0	0
o is Literal	1	1	10
val+type OK	13	13	13

GPT 4

Incorrect Property Usage

- A mapping for the (wrong) property editor was generated once by both models
- The (wrong) producer property was mapped
  - six times by Claude-3.
  - one time by GPT-4

	editor	producer	runtime
p is used	1	6	0
p fecq. OK	25	20	26
o is Object	1	6	0
o is Literal	0	0	0
val+type OK	26	26	26

Claude 3

	editor	producer	runtime
p is used	1	1	10
p fecq. OK	12	12	3
o is Object	0	0	0
o is Literal	1	1	10
val+type OK	13	13	13

GPT 4

Incorrect Property Usage

- A mapping for the (wrong) property editor was generated once by both models
- The (wrong) producer property was mapped
  - six times by Claude-3.
  - one time by GPT-4
- The (wrong) runtime property was only used by GPT4 <u>never wrongly by Claude-3</u>

	editor	producer	runtime
p is used	1	6	0
p fecq. OK	25	20	26
o is Object	1	6	0
o is Literal	0	0	0
val+type OK	26	26	26

Claude 3

	editor	producer	runtime
p is used	1	1	10
p fecq. OK	12	12	3
o is Object	0	0	0
o is Literal	1	1	10
val+type OK	13	13	13

GPT 4

Conclusion

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future
  - Claude-2.1 GPT3.5 Gemini Pro could not tackle this task

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future
  - Claude-2.1 GPT3.5 Gemini Pro could not tackle this task
- Open challenges in RDF syntax generation

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future
  - Claude-2.1 GPT3.5 Gemini Pro could not tackle this task
- Open challenges in RDF syntax generation
  - Prefixes, . , ; usage to separate Triple (Elements)

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future
  - Claude-2.1 GPT3.5 Gemini Pro could not tackle this task
- Open challenges in RDF syntax generation
  - Prefixes, . , ; usage to separate Triple (Elements)
- Open challenges in RML mapping generation

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future
  - Claude-2.1 GPT3.5 Gemini Pro could not tackle this task
- Open challenges in RDF syntax generation
  - Prefixes, . , ; usage to separate Triple (Elements)
- Open challenges in RML mapping generation
  - Input Source, JSON path query, ontology understanding (object vs datatype property)

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future
  - Claude-2.1 GPT3.5 Gemini Pro could not tackle this task
- Open challenges in RDF syntax generation
  - Prefixes, . , ; usage to separate Triple (Elements)
- Open challenges in RML mapping generation
  - Input Source, JSON path query, ontology understanding (object vs datatype property)
- Many Directions for Future Work

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future
  - Claude-2.1 GPT3.5 Gemini Pro could not tackle this task
- Open challenges in RDF syntax generation
  - Prefixes, . , ; usage to separate Triple (Elements)
- Open challenges in RML mapping generation
  - Input Source, JSON path query, ontology understanding (object vs datatype property)
- Many Directions for Future Work
  - Complex Mappings, Formats, Different Domain, Fine tuning, Evaluation Method, Other Tools

Conclusion

- Claude-3-Opus and GPT-4 showed promising results for RML generation in the future
  - Claude-2.1 GPT3.5 Gemini Pro could not tackle this task
- Open challenges in RDF syntax generation
  - Prefixes, . , ; usage to separate Triple (Elements)
- Open challenges in RML mapping generation
  - Input Source, JSON path query, ontology understanding (object vs datatype property)
- Many Directions for Future Work
  - Complex Mappings, Formats, Different Domain, Fine tuning, Evaluation Method, Other Tools

Thank You!

Contact:
hofer@informatik.uni-leipzig.de

InfAI® Institute for Applied Informatics
UNIVERSITÄT LEIPZIG
ScaDS.AI DRESDEN LEIPZIG