January 16

Hollywood Ontology

2023

[Type the abstract of the document here. The abstract is typically a short summary of the contents of the document. Type the abstract of the document here. The abstract is typically a short summary of the contents of the document.]

By Maxwell Okumu

INTRODUCTION

The entertainment industry is a vast and dynamic field, with a wealth of information on movies, TV shows, actors, directors, and related entities. The nature of this information is that it is scattered across numerous sources on the internet, making it a burdensome task to try and get insight from a unified coherent view of the entirety of this data. In this project, I aimed to create a Hollywood ontology to overcome this issue. The ontology is a structured, machine-readable representation of knowledge about the entertainment industry that can be queried and visualized to gain insights and answer complex questions.

PROBLEM STATEMENT

The problem the Hollywood ontology aims to solve is the lack of a consistent and unified view of the entertainment industry. The information about movies, actors, directors, and awards is scattered across various sources, such as IMDB, Wikipedia, and other websites. This makes it difficult to access and understand the information, and to answer complex questions about the relationships between different entities in the industry.

SOLUTION OVERVIEW

The proposed solution to this problem is a Hollywood ontology that provides a structured, machine-readable representation of knowledge about the entertainment industry. The ontology is based on the RDF data model and uses the OWL ontology language. It defines classes such as Movie, Actor, Director, Studio, and Award, and properties such as title, directedBy, and starredIn. The ontology also defines constraints and rules to ensure data quality and consistency.

DATA MODEL

The Hollywood ontology is based on the RDF data model, which is a flexible and extensible data model that is well-suited for representing knowledge. The RDF data model consists of triples, which are statements that describe the relationships between entities. Each triple has a subject, a predicate, and an object. In this ontology, the subjects are the entities, such as movies, actors, and directors, the predicates are the properties, such as title and directedBy, and the objects are the values of the properties.

The Hollywood ontology classes are defined as follows:

- Movie class: representing movies, with properties such as title, year, and rating,
- Actor class: representing actors, with properties such as name and birthdate
- Director class: representing directors, with properties such as name and birthdate
- Studio class: representing studios, with properties such as name and location
- Award class: representing awards, with properties such as name and year

The ontology also defines object properties such as directedBy, starredIn, producedBy and data properties such as title, year, rating, etc.

Example of movie class

Class	**Data Property**	**Data Type**	**Description**
Movie	title	string	The title of the movie
	year	int	The release year of the movie
	rating	float	The rating of the movie (e.g. 8.5)
	genre	string	The genre of the movie

	certificate	string	The certificate of the movie

DATA SOURCES

Our Hollywood ontology is populated with data from multiple sources such as IMDB, Wikipedia and other websites. I used web scraping techniques to extract the data from these sources and transformed it into the RDF format. The data sources were chosen based on their relevance and reliability. They are as follows:

1. IMDb (Internet Movie Database) - a database of information related to films, television shows, and video games, including cast and crew, plot summaries, and reviews.
2. Rotten Tomatoes - a website that aggregates reviews from film and television critics to provide a score for a movie or show.
3. Box Office Mojo - a website that tracks box office revenue for movies and provides information on budget and release dates.
4. The Movie Database (TMDb) - a community-driven database for movies and TV shows, with information like release dates, cast, images, and trailers.

PROCESS WORKFLOW:

The process workflow followed to create the Hollywood ontology is as follows:

1. Identifying the relevant data sources and Scraping data from them using a web scraping tool called Scrapy. This scrapes data such as movie titles, release dates, cast and crew, plot summaries, and reviews.

   Here is snippet of Python code I used that uses the Scrapy library to scrape the IMDb website for movie titles and their respective URLs:

   ```
   import scrapy class IMDbSpider(scrapy.Spider): name = "imdb" start_urls = [
   'https://www.imdb.com/search/title?genres=action&start=1', ] def parse(self,
   ```

```
response): for movie in response.css('div.lister-item'): yield { 'title':
movie.css('h3 a::text').get(), 'url': movie.css('h3 a::attr(href)').get(),
'year': movie.css('span.lister-item-year::text').get(), 'rating':
movie.css('div.ratings-imdb-rating strong::text').get(), 'votes':
movie.css('span.sort-num::text').get(), 'certification':
movie.css('span.certificate::text').get(), 'genres':
movie.css('span.genre::text').getall() } next_page = response.css('a.next-
page::attr(href)').get() if next_page is not None: yield
response.follow(next_page, self.parse)
```

The results are availed in the *dataset_imdb-scrapper.csv* file in the hollywood ontology github repository.

2. Pre-processing the scraped data by cleaning and normalizing it. This entailed removing special characters, duplicates and standardizing the format of movie titles, names of actors and directors.

3. Using named entity recognition (NER) to extract entities such as movie titles, actors, directors, and other individuals involved in the production of films and television shows. SpaCy, a pre-trained model came in handy for this process.

4. Relation extraction to identify relationships between entities. OpenIE, a pre-trained relationship extraction model came in handy for this.

5. Defining the classes, properties, and constraints of the ontology using the OWL ontology

4. Mapping the extracted data to the classes and properties of the ontology using RDF triples

5. Use Protege to visualize the ontology and perform reasoning and inferencing

6. Testing the ontology using SPARQL queries to ensure it is functioning as expected.

7. Validate the ontology by testing it against sample queries to ensure it can be used to answer questions about the Hollywood domain.

Example query to answer the question: "Which movies directed by Christopher Nolan won an award?"

```
SELECT ?movie WHERE { ?movie a :Movie. ?movie :hasDirector :ChristopherNolan.
?movie :hasAward ?award. FILTER NOT EXISTS { ?award a :NominatedAward } }
```

This query retrieves all the movies that have Christopher Nolan as a director and have an award, but not a nominated award. It uses the triple pattern to match all triples where the subject is a movie, the predicate is "hasDirector" and object is ChristopherNolan and the predicate is "hasAward" and object is ?award and the movie is of type movie, it uses the filter to exclude the movies that have nominated award.

EVALUATION

To evaluate the effectiveness of the Hollywood ontology, I ran a series of SPARQL queries and compared the results to the original data sources. The queries were designed to test the ontology's ability to answer complex questions about the relationships between different entities in the entertainment industry. The results showed that the ontology was able to accurately and efficiently answer the queries, demonstrating its effectiveness.

The full ontology classes with corresponding data and object properties are as follows:

Class	**Data Property**	**Data Type**	**Description**
1.Movie	title	string	The title of the movie
	year	int	The release year of the movie
	rating	float	The rating of the movie (e.g. 8.5)

	genre	string	The genre of the movie
	certificate	string	The certificate of the movie

Class	Data Property	Data Type	Description
2. Actor	name	string	The name of the actor
	birthdate	date	The birthdate of the actor
	birth place	string	The birth place of the actor
	occupation	string	The occupation of the actor
	years_active	string	The years active of the actor

Class	Data Property	Data Type	Description
3.Director	name	string	The name of the director
	birthdate	date	The birthdate of the director
	birth place	string	The birth place of the director
	occupation	string	The occupation of the director
	years_active	string	The years active of the director

Class	Data Property	Data Type	Description
4.Studio	name	string	The name of the studio
	location	string	The location of the studio
	founded_year	int	The founded year of the studio
	area_of_activity	string	The area of activity of the studio
	parent_company	string	The parent company of the studio

Class	Data Property	Data Type	Description
5.Award	name	string	The name of the award
	year	int	The year the award was given
	category	string	The category of the award

Class	Data Property	Data Type	Description
	recipient	string	The recipient of the award
	presenter	string	The presenter of the award

References:

- RDF Data Model: https://www.w3.org/RDF/
- OWL Ontology Language: https://www.w3.org/OWL/
- Web Scraping Techniques: https://towardsdatascience.com/web-scraping-tutorial-with-python-beautiful-soup-8a79c5b7b5d2
- RDF stores: https://jena.apache.org/documentation/rdfstore/index.html
- Protege: https://protege.stanford.edu/
- SPARQL: https://www.w3.org/TR/rdf-sparql-query/

CONCLUSION:

In conclusion, the Hollywood ontology provides a structured and machine-readable representation of knowledge about the entertainment industry. It allows for easy access and understanding of the information, and can be queried and visualized to gain insights and answer complex questions. The ontology is flexible and extensible, making it easy to update and improve as new data becomes available. The ontology was evaluated using a series of SPARQL queries and showed to be effective in answering complex questions about the entertainment industry. The Hollywood ontology is an example of how ontologies can be used to solve real-world problems and make sense of scattered data.