**Known limitations**

1. **Not answered queries**

Given the time that was available and concerning their weight in the project the following queries could not be answered or was not computationally efficient:

* Dynamic text input

The absence of similarity measures, which will be analyzed further, restricted the handling of complicated queries. Finding similar movies based on the dynamic input, free text, of the user was a query that could not be applied as it required a variety of techniques to prove the similarity with a certain movie.

* Language input

Another reason why the input was fixed, based on a list, was because this application could not handle different languages. The user has to type in English or select from the list as no other language option is available.

* Weight of selection

Another query that couldn’t be implemented as it could perplex the similarity measure that was created and raise the computational time, was the weight of each input in the similarity score to be dynamic. For example, the user determines if the actor is more important to him than the director.

* No input

A query that in theory cannot be answered correctly in this application is when no input is given. The logical way to handle this is by making one of the inputs mandatory or displaying all movies. But to be more minimal was decided only a fixed number of movies to be displayed.

1. **Limitation overcome**

The time limit was the main reason for most of these queries not being answered. As mentioned, their contribution to the result and the difference that could make was not so big and therefore, they were not implemented.

Given more time, the query about dynamic input could be handled with a more complex similarity measure. At first, the input will be stored and filtered with the existing movies to look for a match. If does not exist any, then every character of the input should be compared with every character of all movies and calculate how similar each movie is with the input. Finally, the movie or movies with the highest score will be considered the desired ones, and the similarity measure for them will be applied to find the recommendation.

For the language, the only thing that should be done is for every data to be translated into a variety of languages to allow multilingual inputs. But consequently, this will increase exponentially the computational time.

As for the weighting of inputs in the similarity score, again, the similarity measure will be more complicated, with dynamic weight inputs that will be given by the user.

1. **RDF & Semantic Web Technologies**

Even though RDF Data models and Semantic Web Technologies offer significant advantages in data interoperability and knowledge representation, they face certain limitations.

* Complexity

RDF’s triple-based structure and SPARQL queries are not well known and require specialized knowledge for complex queries to be crafted and executed without errors.[3] Their structure (subject, predicate, object) restricts the exact representation of relationships like temporal dependencies, making the illustration of some more perplexed real-world scenarios very difficult**.**[2]

* Scalability

Querying and reasoning over big datasets, thus large RDF graphs, and especially on complex queries can be computationally very expensive. These high processing times can lead to performance issues as the model’s flow will be corrupted. But this problem is known, and researchers are trying to solve this issue with specialized storage systems and efficient query engines to limit this performance bottleneck.[1][4][5]

* Privacy and Security

As is observed in this project and in general, Semantic Web and RDF lack security and privacy. They are mostly designed for open and transparent sharing of data with the management of permissions to secure data being a little complicated.[3]

1. **Triple reduction**

On the one hand, Movie Databases are a bit restricted in terms of properties as they have a limited amount of properties that can have, but, on the other hand, not all movies have all the properties. This happens because of different time eras, with different wanted properties, and knowledge. For example, a movie that was produced in 1940 will not have a property for VFX as the new ones, or that years they might not care about the box office.

Given that, filtering the properties of movies based on the most known and the one we needed the most was a way to reduce the total number of triples, without losing any important info for the implementation of the project.

Another observation that helped in reducing the number of triples was that some movies had the same properties in different languages. As mentioned, because it was decided the default and only language to be English, all triples that were in different languages were filtered, reducing their number drastically.

References:

[1] Mohammed, Wria & Jumaa, Alaa. (2022). A SURVEY ON USING SEMANTIC WEB WITH BIG DATA: CHALLENGES AND ISSUES. Harbin Gongye Daxue Xuebao/Journal of Harbin Institute of Technology. 54. 93-103.

[2] Dean, M., & Schreiber, G. (2004, February 10). *Owl web ontology language reference*. OWL Web Ontology Language Reference. <https://www.w3.org/TR/owl-ref>

[3] Kirrane, Sabrina, Villata, Serena, and d’Aquin, Mathieu. ‘Privacy, Security and Policies: A Review of Problems and Solutions with Semantic Web Technologies’. 1 Jan. 2018 : 153 – 161.

[4] Le-Tuan A, Hayes C, Hauswirth M, Le-Phuoc D. Pushing the Scalability of RDF Engines on IoT Edge Devices. Sensors (Basel). 2020 May 14;20(10):2788. doi: 10.3390/s20102788. PMID: 32422961; PMCID: PMC7284853.

[5] Pelgrin, O. (2024). Expressive Querying and Scalable Management of Large RDF Archives - Source code and Experiments (1.0). Zenodo. https://doi.org/10.5281/zenodo.11517199