Proceedings of the 13th Conference on Language Resources and Evaluation (LREC 2022), pages 1297–1306
Marseille, 20-25 June 2022

A Multi-source Graph Representation of the Movie Domain for Recommendation Dialogues Analysis

Antonio Origlia[1], **Martina Di Bratto**[1], **Maria Di Maro**[1], **Sabrina Mennella**[2]
[1]University of Naples Federico II, [2]University of Catania
{antonio.origlia, martina.dibratto, maria.dimaro2}@unina.it, sabrina.mennella@phd.unict.it

Abstract

In dialogue analysis, characterising named entities in the domain of interest is relevant in order to understand how people are making use of them for argumentation purposes. The movie recommendation domain is a frequently considered case study for many applications and by linguistic studies and, since many different resources have been collected throughout the years to describe it, a single database combining all these data sources is a valuable asset for cross-disciplinary investigations. We propose an integrated graph-based structure of multiple resources, enriched with the results of the application of graph analytics approaches to provide an encompassing view of the domain and of the way people talk about it during the recommendation task. While we cannot distribute the final resource because of licensing issues, we share the code to assemble and process it once the reference data have been obtained from the original sources.

Keywords: graph databases, movies domain, cross-resource analysis

1. Introduction

Graph databases have gained popularity, in recent years, for their capability to efficiently represent complex data in an interpretable, flexible format. In particular, graph databases are well-suited in cross-referencing different resources so that information distributed in multiple sources can be analysed in an integrated way. Graph databases have found application in many different fields requiring the management of large datasets characterised by complex interactions among items. In the field of Natural Language Processing, Word Sense Disambiguation (Yao et al., 2017), Knowledge Base Collection (Yu et al., 2020) and Fraud Detection (Srivastava and Singh, 2018; Stray, 2019) are strongly investigated areas. Also, node embedding techniques have been significantly improved to capture latent network semantics to support machine learning tasks (Chanpuriya et al., 2020; Yang et al., 2020).
In this sense, the movies domain represents a well studied case of interest for a number of different tasks, from dialogue analysis in recommendation (Lee and Choi, 2017) to movie revenues prediction (Zhou et al., 2019) and movie rating prediction (Forouzandeh et al., 2021). As there are many different approaches and points of view about this specific domain, we present a data processing pipeline to integrate, in a single, graph-based, resource, the information contained in some of the most relevant resources that are available online. Also, we apply graph analysis techniques to enrich the obtained network with latent information extracted from the final graph structure. More in detail, the procedure can be summarised in three steps:

- Data collection
- Graph database assembling
- Data enrichment

The main goal of this work is to support investigation in the field of argumentation based dialogue, for which a general theoretical framework is still missing (Prakken, 2018). To fill this gap, the database represents an instrument to analyse argumentation-based dialogues strategies from a linguistic point of view while providing a deep *context* of the dialogue itself. The presented database can serve as a model structure to be used for the development of more advanced recommender dialogue systems given that, the movie application domain allows the extraction and the use of relevant information for argumentation purposes (Section 5). The paper is organised as follows: Section 2 covers the first step, describing the contents of the considered resources and the way they were accessed. Section 3 describes how these were cross-referenced and organised into a single, graph-based, database, summarising previous work presented in (Di Bratto et al., 2021) and extending it. Section 4 describes the data enrichment procedures applied to extract latent information from the graph structure. In Section 5, both the linguistic and the technological applications for the resource will be discussed. While we are not able to share the final database, as we cannot redistribute data coming from some of the considered sources, the Python source code to generate it, once the data are obtained from the official sources, is freely available[1] together with the results of the graph analysis algorithms we used.

2. Data sources

Automatically collecting and organising large amounts of data is relatively easy, nowadays, given the availability of Linked Open Data (LOD). Graph databases, in particular, provide the necessary flexibility to cross-reference different resources, enabling dialogue analy-

[1]https://github.com/antori82/MS_MovieGraph

sis in relationship with domain knowledge representation. From a linguistic point of view, graph databases have the potential to unlock the extraction of latent knowledge that is not immediately accessible from the single resources. Specifically, concepts coming from graph theory and applications, like Pagerank scores and node embeddings, can help identify regularities in the dialogues that can, later, instruct the design of dialogue systems. The knowledge base for the movies domain is built by collecting data from different sources and organising it so that cross-referencing is possible, whose importance is demonstrated for argumentation purposes (Section 3). The included resources are described in the rest of this Section.

2.1. Internet Movie Database

The Internet Movie Database (IMDb) is the most popular dabatase for movie, TV, and celebrity contents[2]. It includes more than 8 million films and programmes, and about 10 millions actor names. Besides the information regarding the movies, registered users can also submit a rating (on a scale of 1 to 10) per each movie. In total, there are nowadays more than 1 million ratings.
In different studies, such as (Herr et al., 2007), the importance of studying and applying the IMDb database has been underlined. Firstly, the richness of the database allows a wide variety of data analyses. Secondly, the dataset is well structured and linked to other datasets, like Wikidata, so the need of semantic matching techniques, potentially causing errors, is limited.

2.2. Wikidata

IMDb does not provide an ontological view of movies and it does not report information about awards won by people and movies. These, however, are important aspects to consider when using domain items for argumentation purposes. Such information is found in Wikidata, which also contains the IMDb ids (P345) for both movies and persons. Connecting Wikidata to IMDb is straightforward, given the already existing alignment. The *award_received* (P166) relationship in Wikidata connects movies and awards while also providing optional qualifiers to further detail the relationship. The qualifier we consider for *award_received* is *winner*, connecting the relationship to one or multiple people.

2.3. Inspired

The Inspired corpus (Hayati et al., 2020) is a dataset containing 1001 recommendation dialogues of two-paired crowd-workers who chat in a natural setting in English. The conversations take place between a recommender and a movie seeker. In order to achieve their communicative goal, the recommenders have to investigate the seekers' tastes and propose the most suitable movie. The corpus presents a total of 35,811 utterances and a average of turns per dialogue equal to 10.73 since recommenders are asked to continue the chat for a minimum of 10 turns. Furthermore, the dataset provides manual annotations for the recommenders' utterances. The scheme is based on sociable recommendation strategies used by the recommenders to persuade the seekers and encourage them to accept the proposed movie. Regarding the recommenders' strategies, they are divided in two main categories: preference elicitation strategies and sociable strategies. The two categories could be respectively associated with the two typical phases of the recommendation dialogue, i.e., user information gathering and movie recommendation (Hayati et al., 2020). Through the analysis of the connection between the use of sociable strategies and the recommendation results, Hayati and their colleagues show how sociable strategies have a positive impact on the acceptance of recommendation and how those strategies could improve the dialogue quality. As an extension of this work, sociable strategies are cross-referenced with domain items characterisation and provide a deeper understanding of the dialogue flow (Di Bratto et al., 2021).

2.4. Movielens

Data concerning user evaluations about movies are not taken from the IMDb's rating data but from the Movielens 25M dataset (Harper and Konstan, 2015). This contains 25000095 ratings and 1093360 tags across 62423 movies. The data were created by 162541 users between 1995 and 2019. All selected users had rated at least 20 movies and had been randomly selected. Movielens is publicly available and it represents a widely used dataset for movie recommendation tasks with machine learning.

3. Graph representation

To represent the full set of data, we adopt Neo4J (Webber, 2012): an open source graph database manager that has been developed over the last 16 years and applied to a high number of tasks related to data representation (Dietze et al., 2016), exploration (Drakopoulos et al., 2015) and visualisation (Jiménez et al., 2016). Neo4j is characterised by high scalability and ease of use. Differently from other graph-based approaches, like the ones based on RDF, it uses data structures that are designed for performance speed and optimised for graph traversal operations. In general terms, whereas approaches based on RDF are designed for compatibility and general purpose knowledge representation, graph databases are more application-oriented. Concerning recommender systems, Neo4j has been used in (Sansonetti et al., 2019) on social media data coming from Facebook and cultural heritage data coming from DBPedia and Europeana. Neo4j also integrates a machine learning library[3] optimised for graph data analysis. This includes different kinds of functions to com-

[2] http://www.imdb.com/

[3] https://neo4j.com/docs/graph-data-science/current/

pute different measures to characterise graph nodes, like centrality and neighbourhood similarities.

The first step of the graph building procedure consists in importing the data provided by IMDb, which is the most structured data source available for the specific domain. The main node labels, imported after this step, are MOVIE and PERSON. Moreover, for the MOVIE nodes, genres are also represented as GENRE nodes (e.g. WESTERN, THRILLER, etc...), linked by a HAS_GENRE relationship. MOVIE and PERSON nodes are linked by WORKED_IN, WROTE and DIRECTED relationships, mainly. To introduce textual description of available movies in the database, plots and synopses written by multiple users about that movie were also collected from IMDb. For each textual description, a PLOT node is created and connected to the MOVIE node it refers to by a HAS_PLOT relationship. The text of the plot is stored as a *text* property in the PLOT nodes. Node count statistics for this first part of the procedure are reported in Figure 1.

The second part of the procedure covers dialogue data from the Inspired corpus. Since Inspired is constituted by a sequence of turns between two subjects acting either as an information seeker or as a domain expert recommender, the most natural way to represent dialogue history in a graph form is by a nodes chain. Therefore, each turn is represented by an UTTERANCE node also having a RECOMMENDER/SEEKER label representing who stated the sentence. Since there are no returning users in Inspired, there is no need to model them as separate nodes. UTTERANCE nodes expressing specific intents are marked with labels corresponding to the Inspired sociable strategies and are linked using FOLLOWED_BY relationships to keep track of the turns sequences. Connecting Inspired to the data collected from the web is performed using text matches between sub-strings marked as named entities in the original dataset (e.g. movies, people, etc...) and the name fields collected from IMDb and Wikidata. At this time, potential ambiguities are introduced in the graph due to homonyms: these are later solved during graph analysis, as discussed in Section 4. Links between UTTERANCEs and named entities (MOVIEs and PERSONs) are represented using REFERS_TO relationships. Statistics concerning the nodes imported after the second step are reported in Figure 2.

At the end of this step, it is possible to analyse the features of the items involved in the dialogue to demonstrate the importance of cross-referencing resources in a graph structure. As an example, we analyse the release years of the MOVIE items named in the Inspired dialogues, This analysis points out a strong bias towards recently released MOVIEs, as shown in Figure 3. This is an interesting note as it underlines how strong patterns can be found in attribute values distributions, other than in structural relationships. While there are machine learning approaches that are able to capture these patterns, like GraphSage embeddings (Hamilton et al., 2017), many graph-based approach mainly rely on structural patterns only and may miss this kind of information. From the point of view of dialogue analysis, this is an indication that data collection time should be taken into account as a normalisation factor when considering the reason why named items were selected to support speech acts.

The third part of the procedure covers the integration of the Movielens dataset. This contains the IMDb identifiers for the movies ratings are referred to so the import step is straightforward. A MOVIELENSUSER node is created for each user in the dataset and a RATED relationship with a MOVIE node for each reported rating in the dataset. RATED relationships have a *score* property to represent the actual rating.

The database schema resulting from the data collection procedure is summarised in Table 1, showing node and relationship properties, and in Table 2, showing source and target nodes for the different relationships. An extract of the obtained structure is shown in Figure 4.

4. Data enrichment

After organising data coming from different sources in a single structure, we proceed to enrich the database using graph-specific algorithms to capture latent information.

4.1. Plot similarities

To compute textual similarities among movies using PLOT nodes, we only consider the ones related to MOVIE nodes that are not episodes of long-running series nor TV shows or documentaries, obtaining 1085742 plots. Also, we consider the results shown in (Ranashinghe et al., 2019), which have demonstrated that, on the sentence similarity task, stacked embeddings composed of ELMo (Peters et al., 2018) and BERT embeddings outperformed other solutions when results were weighted by Smooth Inverse Frequency (SIF) (Arora et al., 2017). In this specific case, we compute plots semantic representations using stacked ELMo and RoBERTa (Liu et al., 2019) embeddings. The procedure consists of three steps:

- Weighted document embeddings computing
- Principal Components Analysis
- Embeddings re-evaluation

First of all, sentence embeddings are computed, using pre-trained networks, for each sentence s in each document, as the weighted average of word embeddings found in the sentence. Weights are computed from the word frequencies, expressed as the word occurrence probability $p(w) : w \in W$ where W are the words in the entire corpus. Given a constant value a, which we set to 0.001, weights are computed as $a/(p(w) + a)$, so that more frequent words are weighted less. The weighted sentence embedding v_s is, therefore, obtained as

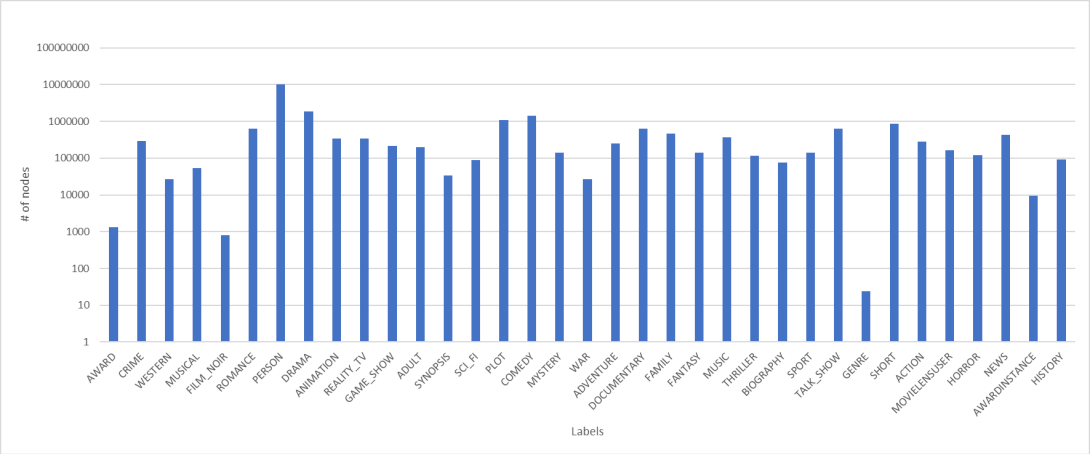

Figure 1: Number of nodes, per category, generated after the first import step. Note that a single node may have multiple labels. Given the strong presence of PERSON nodes, in the database, we use a logarithmic scale.

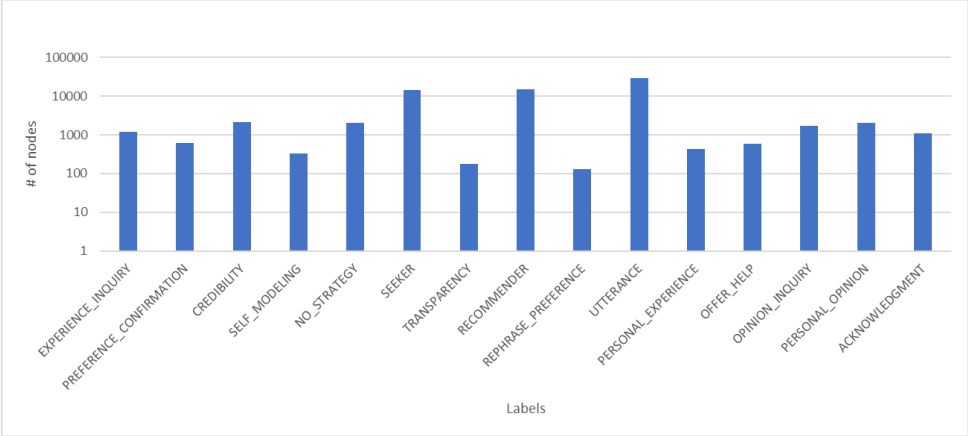

Figure 2: Number of nodes, per category (as reported in (Hayati et al., 2020)), generated after the second import step. Note that a single node may have multiple labels. Given the strong presence of UTTERANCE nodes, in the database, we use a logarithmic scale.

Label	**Properties**
MOVIE	imdbID, primaryTitle, startYear, endYear, runtimeMinutes, wfrpembedding, frpembedding, pageRank
PERSON	imdbID, primaryName, birthYear, deathYear, wfrpembedding, frpembedding, pageRank
PLOT	text
AWARD	primaryName
AWARDINSTANCE	-
UTTERANCE	text
MOVIELENSUSER	frpembedding
WORKED_IN	category, job, role
EPISODE_OF	seasonNumber, episodeNumber
RATED	score
SIMILAR_TO	degree
AWARDED_FOR	-
AWARDED_TO	-

Table 1: Node and relationship properties in the database.

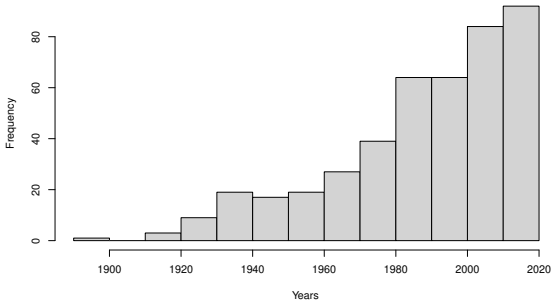

Figure 3: Release years of the movies people included in Inspired talked about. The bias towards recent movies is clearly visible.

Label	Relationship	Label
MOVIE	HAS_PLOT	PLOT
PLOT	SIMILAR_TO	PLOT
PERSON	KNOWN_FOR WORKED_IN DIRECTED WROTE	MOVIE
AWARDINSTANCE	IS_A	AWARD
AWARDINSTANCE	AWARDED_TO	PERSON
AWARDINSTANCE	AWARDED_FOR	MOVIE
UTTERANCE	FOLLOWED_BY	UTTERANCE
UTTERANCE	REFERS_TO	PERSON MOVIE
MOVIELENSUSER	RATED	MOVIE

Table 2: Nodes and relationships between nodes in the final database.

$$v_s = \frac{1}{|s|} \sum_{w \in s} \frac{a}{a + p(w)} v_w \quad (1)$$

Document embeddings v_d are obtained by considering the average of the sentence embeddings $v_s : s \in d$ and $d \in D$ where D is the set of all documents. After computing document embeddings, we consider the

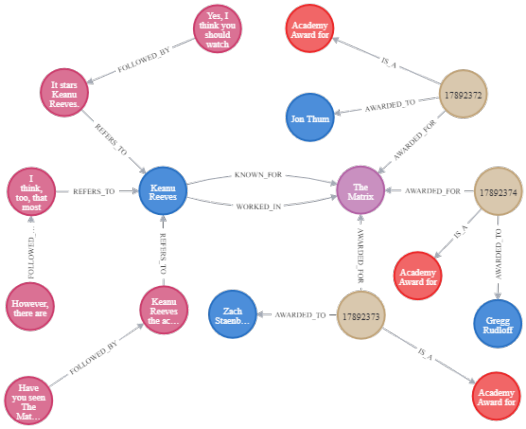

Figure 4: An extract of the graph structure, including MOVIEs (purple), PERSONs (blue), AWARDINSTANCEs (brown) of AWARDs (red) and UTTERANCEs (pink).

X matrix whose rows are the v_d embeddings. The final embedding $\bar{v}_d$ is obtained by subtracting, from each v_d, the first principal component of the X matrix. This removes, from the semantic representation, elements that are common to all the documents, leaving the traits with actual discriminative power in the data.

For each pair of document embeddings, similarity is computed as the cosine of the two SIF-weighted embeddings. A SIMILAR_TO relationship is, then, established in the database between two PLOTs if the cosine value is higher than 0.8. The SIMILAR_TO relationship has a *degree* property containing the cosine between the two SIF-weighted stacked embeddings. As an example of the result of this procedure, we consider the case of the movie *Black Swan*. By extracting the top five MOVIEs that share similar topics in their PLOTs, the database reports the MOVIEs *The Red Shoes*, *Jeanne Eagels*, *Flashdance*, *Funny girl* and *Stage struck*, all sharing the dance topic. For the case of *The Matrix*, instead, the top five is *"Control, Alt, Delete"*, *Elysium*, *Captain America*, *Limitless* and *"I, Robot"*, sharing sci-fi or superhuman topics. After the data enrichment steps, the final size of the database consists of 110 million nodes and approximately 6 billion relationships.

4.2. PageRank

While linking Inspired to domain knowledge, the presence of homonyms can be cause of ambiguity, as shown in the previous Section. In such cases, participants refer to specific items sharing their name with other ones without providing further specification. In these cases, the *importance* of a single node among all the potential ones is significantly stronger than all the others so that ambiguities are only apparently there. To solve these apparent ambiguities, the Pagerank algorithm can be used. PageRank assigns an *importance* value to items in the graph as a measure of their relevance in the graph, in terms of connections. Introduced by (Page et al., 1999), PageRank was designed for Information Retrieval and was widely used in web searches. As an example of the application of PageRank in our framework, an UTTERANCE extracted from the Inspired corpus is shown in Figure 5. In this example, the actor *Chris Evans* is mentioned. There are multiple items representing people named Chris Evans, in the database, but only one item has a PageRank score significantly higher than all the others, thus removing the ambiguity. PageRank scores are, thus, can be essential to support entity recognition based on its relevance in a specific domain.

4.3. Node embeddings

To provide context-dependent representations of node semantics, we apply the extended version of the Fast Random Projection algorithm implemented in the Neo4j Graph Data Science (GDS) Library. Although based on the original work presented in (Achlioptas, 2003), GDS extends it by supporting the use of

Figure 5: Graph showing how an UTTERANCE node (in pink) can be related to more than one PERSON nodes (in blue) having the same string label (i.e., *Chris Evans*). Such cases can be disambiguated through the PageRank algorithm.

node properties and relationship weights. In (Chen et al., 2019), FastRP was compared with other well-known node embeddings methods like, among others, node2vec (Grover and Leskovec, 2016) and DeepWalk (Perozzi et al., 2014). Results showed that machine learning approaches trained using FastRP embeddings reached competitive performances on multiple classification tasks while critically reducing computation time. FastRP is based on the Johnson-Lindenstrauss lemma, stating that a set of points distributed in a high-dimensional space can be embedded in a space of significantly lower dimension while, at the same time, preserving distances among points. The original algorithm first assigns a random vector e_n^0 to each n node in the graph. For a given number of iterations k, an intermediate embedding is computed as the average of the neighbouring embeddings as

$$e_n^i = avg(e_m^{i-1}) \tag{2}$$

where m ranges over the nodes' neighbours. The final node embedding e_n is obtained as the weighted average of the intermediate embeddings, normalised by their L2 norm as

$$e_n = \sum_{i=0}^{k} w_i \cdot \frac{e_n^i}{\sqrt{e_n^i \cdot e_n^i}} \tag{3}$$

The number of iterations, together with the weights sequence $[w_0, \ldots, w_k]$ defines the radius of the neighbourhood considered in the final embedding and the influence of the nodes at each distance.

The GDS implementation of FastRP extends the original algorithm to weighted graphs by computing, for each e_n^i, the *weighted* average of the neighbours embeddings from the preceding iteration. Also, node properties are taken into account during the generation of the initial random vector. Specifically, as a first step, each node property is associated with randomly generated vectors. Then, e_n^0 is generated as the concatenation of two sub-vectors:

- A random vector as in the original algorithm
- A linear combination of the property vectors, using the property values as weights

The generation of the initial random vectors is defined by a sparse random projection matrix (Chen et al., 2019) defined as

$$R_{ij} = \begin{cases} \sqrt{s} & \text{with probability } \frac{1}{2s} \\ 0 & \text{with probability } 1 - \frac{1}{s} \\ -\sqrt{s} & \text{with probability } \frac{1}{2s} \end{cases} \tag{4}$$

where s, in the GDS implementation, depends on the embedding dimension d and on the node degree d_j. To normalise with respect to node degrees, the algorithm applies a normalisation matrix L_{ij} defined as

$$L_{ij} = \left(\frac{d_j}{2m}\right)^{\beta} \tag{5}$$

where m is the number of edges in the graph and β is a normalisation factor. Given the adjacency matrix S and the degree matrix D, the transition matrix for the nodes in the graph is computed as $A = D^{-1}S$. As β goes to infinity, we obtain

$$A_{ij}^k = \left(\frac{d_j}{2m}\right) \tag{6}$$

which describes the probability of reaching the j-th node from the i-th node in k steps of random walk. Moreover, in the GDS implementation, β is also used to compute s as

$$s = (d_j)^{\beta} \frac{\sqrt{3}}{d} \tag{7}$$

Since rating prediction is a standard task for the application of node embeddings to the considered domain, to avoid data leakage from the general domain to the users included in Movielens, FastRP embeddings have been computed over separate graph projections. For the first one, we consider MOVIE and PERSON nodes that were RATED by MOVIELENSUSERs to reduce the size of the domain so that it was tractable in terms of memory. All MOVIE and PERSON items to which the Inspired corpus UTTERANCEs REFER_TO are included in this subset so cross-resource consistency is kept. Linked AWARD, AWARDINSTANCE and GENRE nodes were also included. SIMILAR_TO relationships among PLOTs are mapped directly between MOVIE nodes to make the projection more compact. Given the observed relevance of the MOVIEs release year reported in Section 3, this property is imported. Both weighted and unweighted versions of FastRP embeddings were computed where in the weighted

version, the *degree* property of the SIMILAR_TO relationships is considered. Since we are mainly interested in computing embeddings for MOVIE and PERSON nodes, to adequately support the computation of random walks originating in these nodes, the graph projection removes the orientation of relationships established between MOVIEs and PERSONs, inverts the orientation of relationships that are incoming to these nodes and keeps the orientation of their outgoing relationships. In the first projection, we set the embedding dimension to 32 where the last 4 components depend on the *startYear* node property. The algorithm is limited to three iterations so that third degree neighbours are considered. This is because the longest chain of interest we identified consisted of the nodes sequence describing an AWARDINSTANCE which IS_A AWARD and was AWARDED_TO a PERSON who WORKED_IN a MOVIE.

The second graph projection concerns MOVIELENSUSERs and MOVIEs, together with the RATED relationships linking them. In order to support machine learning applications, 80% of the RATED relationships are marked for training while the rest are left as a test set. FastRP embeddings are computed considering only training relationships. In this case, only weighted embeddings are computed, considering the *score* relationship from the RATED relationships as a weight. This is because the presence of a RATED relationship may express both a positive or a negative view on the MOVIE. In this projection, we computed embeddings of length 32 but there were no properties taken into account. Two iterations were performed to consider up to second-degree neighbours.

Evaluating embeddings quality is a complex topic that is usually investigated using machine learning techniques for prediction tasks. While the database structure has been designed to support the rating prediction task using the Movielens database, as described before, this is beyond the scope of this paper. For this reason, we provide 3D data visualisations of FastRP embeddings obtained using the T-SNE algorithm (Van der Maaten and Hinton, 2008) on the first projection. For all the considered cases, we used the same, empirically chosen, hyperparameters for the T-SNE algorithm. Specifically, we set perplexity to 25, learning rate to 10 and number of iterations to 200. All vectors were normalised to unit length. For demonstration purposes, we also consider embeddings computed while not considering SIMILAR_TO relationships, thus not including PLOT text analysis. Figure 6 shows the 3D projection of the embeddings computed on this situation. Considering the case of the *Star Wars* saga, the represented movies appear to occupy the same space, on the basis of catalographic information only. When considering unweighted SIMILAR_TO relationships, the same movies appear to group together following the different trilogies. An interesting case is represented by *Rogue One*, which is clearly separated from

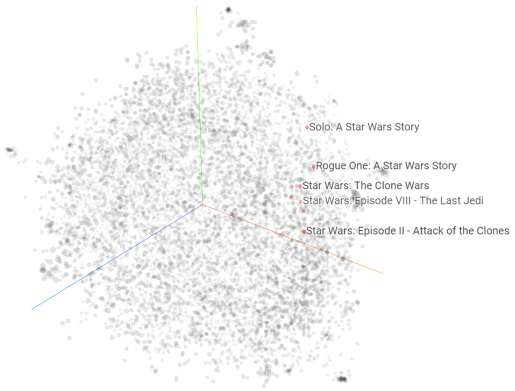

Figure 6: T-SNE 3D projection for the subgraph without PLOT similarities.

Figure 7: T-SNE 3D projection for the subgraph with unweighted PLOT similarities.

all the other movies, as shown in Figure 7. *Rogue One* is included in a movies cluster that, from a deeper inspection, appears to group movies for which text analysis did not find very close neighbours. When using weighted SIMILAR_TO relationships, Figure 8 shows that the separation between *Star Wars* movies is close to the preceding situation, with a slightly stronger separation between movie groups. Whether this difference is of practical use will be evaluated on specific machine learning tasks in future works.
Computing the general domain embeddings is particularly heavy, as the amount of memory needed to compute the graph projection from the database is significant so we provide the results of the procedure for ease of use by the community. Computing embeddings for MOVIELENSUSERs has lower requirements so it is easy to recompute them, should other testing schemas (e.g. cross-validation) be adopted.

5. Applications to argumentation-based dialogue

The presented work can be framed in the field of formal argumentation and, more specifically, it refers to the area of argumentation-based dialogue, which still lacks

Figure 8: T-SNE 3D projection for the subgraph with weighted PLOT similarities.

a solid theoretical framework. Whereas argumentation-based inference assumes a single static and global body of information from which arguments and attacks are constructed, argumentation-based dialogue is dynamic as information is distributed over the dialogue's participants (Prakken, 2018). The presented database, for its structure and format, supports the study of dialogue strategies adopted in the specific case of recommendation dialogues while, at the same time, considering technical aspects concerned with the dialogue systems design. Recommendation dialogues, here represented by the Inspired corpus, are a well studied case of argumentation-based dialogue. In fact, a conversation where two or more interlocutors aim to resolve a conflict of opinion can be considered as a form of persuasion dialogue leveraging on argumentation (i.e., the process of exchanging ideas to establish the truth of a statement). Recommendation dialogues are characterised by two or more participants, usually an expert and a seeker. Typically, the latter discloses their preference while the former tries to make the most suitable recommendation. Once the item has been proposed, the aim of the recommender is to persuade the other participants to accept their recommendation starting, thus, a second dialogue phase characterised by the use of persuasive strategies (Kang et al., 2019; Hayati et al., 2020). This type of interaction could introduce multiple preference conflicts during the persuasion phase due to different opinions and tastes towards a certain item. Moreover, as (Hadoux et al., 2021) point out, persuasion is an important and yet complex aspect of human intelligence. When undertaken through dialogue, the deployment of good arguments, and therefore counterarguments, clearly has a significant effect on the ability to be successful in persuasion.

Exploiting an integrated graph-based structure of multiple sources, as the one we presented in this work, could be fundamental to study argumentation in the case of recommendation processes, since providing more information about a recommended item is a typical approach when providing explanations (Jannach et al., 2020). Argumentation strategies, thus, play a pivotal role in recommendation dialogues and have significant impact in the design of dialogue systems, since showing the reasoning behind a recommendation avoids ineffective feedback from the user and increases their trust and satisfaction (Rago et al., 2018).

Furthermore, in order to argument, a system interacting with a user should collect the information needed to establish a *common ground*, i.e., the "[...] presumed background information shared by participants in a conversation" (Stalnaker, 2002). The proposed resource supports a view of common ground in terms of subgraphs, as shown in (Di Bratto et al., 2021). From a cognitive point of view, these graphs represent different types of Common Ground (Clark, 2015), such as the information shared among speakers of a community (here, IMDb, Wikidata) or the information shared during the past interactions (Inspired or, in a looser sense, Movielens), which becomes the Personal Experience the system can draw upon to choose the right argumentation strategy (Di Bratto et al., 2021). As we have seen, graph databases have a great potential as linguistic tools for dialogue analysis, modelling interaction dynamics and domain knowledge in the same resource. Furthermore, this representation provides a powerful support to dialogue systems through machine learning techniques. With graph databases, both explicit structure and latent knowledge allow to exploit the information stored in the common ground for argumentation purposes to better achieve the communicative goal.

6. Conclusions

We presented the assembling process of a graph database combining multiple data sources into a single structure. Specifically, the database cross-references a corpus of movie recommendation dialogues with domain knowledge collected from different datasets. We also described a data enrichment procedure using text processing techniques and graph data science algorithms to encode complex information into compact representations that can be later used for machine learning tasks. While we cannot distribute the database, as it includes data coming from other sources, we provide the source code to assemble it once the appropriate licences have been obtained. We also provide the results of the data enrichment procedures so they can be directly imported in the database without recomputing the results, some of which required significant computational power. The presented database provides an integrated representation of data coming from very different sources in a way that enables deep analysis using graph data science in a way that was not proposed before. Future work will consist of deploying the database to support the formalisation of a theoretical framework for argumentation-based dialogues, to implement dialogue systems based on this framework and to investigate the effect of alternative graph data science approaches to prediction tasks and latent knowledge extraction.

7. Bibliographical References

Achlioptas, D. (2003). Database-friendly random projections: Johnson-lindenstrauss with binary coins. *Journal of computer and System Sciences*, 66(4):671–687.

Arora, S., Liang, Y., and Ma, T. (2017). A simple but tough-to-beat baseline for sentence embeddings. In *Proceedings of the 5th International Conference on Learning Representations*.

Chanpuriya, S., Musco, C., Sotiropoulos, K., and Tsourakakis, C. (2020). Node embeddings and exact low-rank representations of complex networks. *Advances in Neural Information Processing Systems*, 33.

Chen, H., Sultan, S. F., Tian, Y., Chen, M., and Skiena, S. (2019). Fast and accurate network embeddings via very sparse random projection. In *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*, pages 399–408.

Clark, E. V. (2015). Common ground. In *The Handbook of Language Emergence*, page 328–353. Wiley, Chichester, UK.

Di Bratto, M., Di Maro, M., Origlia, A., and Cutugno, F. (2021). Dialogue analysis with graph databases: characterising domain items usage for movie recommendations. In *Proceedings of Clic.IT*, page to appear.

Dietze, F., Karoff, J., Valdez, A. C., Ziefle, M., Greven, C., and Schroeder, U. (2016). An open-source object-graph-mapping framework for neo4j and scala: Renesca. In *International Conference on Availability, Reliability, and Security*, pages 204–218. Springer.

Drakopoulos, G., Kanavos, A., Makris, C., and Megalooikonomou, V. (2015). On converting community detection algorithms for fuzzy graphs in neo4j. In *Proceedings of the 5th International Workshop on Combinations of Intelligent Methods and Applications, CIMA*.

Forouzandeh, S., Berahmand, K., and Rostami, M. (2021). Presentation of a recommender system with ensemble learning and graph embedding: a case on movielens. *Multimedia Tools and Applications*, 80(5):7805–7832.

Grover, A. and Leskovec, J. (2016). node2vec: Scalable feature learning for networks. In *Proceedings of the 22nd ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 855–864.

Hadoux, E., Hunter, A., and Polberg, S. (2021). Strategic argumentation dialogues for persuasion: Framework and experiments based on modelling the beliefs and concerns of the persuadee. *arXiv preprint arXiv:2101.11870*.

Hamilton, W. L., Ying, R., and Leskovec, J. (2017). Inductive representation learning on large graphs. In *Proceedings of the 31st International Conference on Neural Information Processing Systems*, pages 1025–1035.

Harper, F. M. and Konstan, J. A. (2015). The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, 5(4):1–19.

Hayati, S. A., Kang, D., Zhu, Q., Shi, W., and Yu, Z. (2020). Inspired: Toward sociable recommendation dialog systems. *arXiv preprint arXiv:2009.14306*.

Herr, B. W., Ke, W., Hardy, E., and Borner, K. (2007). Movies and actors: Mapping the internet movie database. In *2007 11th International Conference Information Visualization (IV'07)*, pages 465–469. IEEE.

Jannach, D., Manzoor, A., Cai, W., and Chen, L. (2020). A survey on conversational recommender systems. *arXiv preprint arXiv:2004.00646*.

Jiménez, P., Diez, J. V., and Ordieres-Mere, J. (2016). Hoshin kanri visualization with neo4j. empowering leaders to operationalize lean structural networks. *Procedia CIRP*, 55:284–289.

Kang, D., Balakrishnan, A., Shah, P., Crook, P., Boureau, Y.-L., and Weston, J. (2019). Recommendation as a communication game: Self-supervised bot-play for goal-oriented dialogue. *arXiv preprint arXiv:1909.03922*.

Lee, S. and Choi, J. (2017). Enhancing user experience with conversational agent for movie recommendation: Effects of self-disclosure and reciprocity. *International Journal of Human-Computer Studies*, 103:95–105.

Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., and Stoyanov, V. (2019). Roberta: A robustly optimized bert pretraining approach. *arXiv preprint arXiv:1907.11692*.

Page, L., Brin, S., Motwani, R., and Winograd, T. (1999). The pagerank citation ranking: Bringing order to the web. Technical report, Stanford InfoLab.

Perozzi, B., Al-Rfou, R., and Skiena, S. (2014). Deepwalk: Online learning of social representations. In *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 701–710.

Peters, M. E., Neumann, M., Iyyer, M., Gardner, M., Clark, C., Lee, K., and Zettlemoyer, L. (2018). Deep contextualized word representations. *arXiv preprint arXiv:1802.05365*.

Prakken, H. (2018). *Historical overview of formal argumentation*, volume 1. College Publications.

Rago, A., Cocarascu, O., and Toni, F. (2018). Argumentation-based recommendations: Fantastic explanations and how to find them. In *Proceedings of the Twenty-Seventh International Joint Conference on Artificial Intelligence*, pages 1949–1955.

Ranashinghe, T., Orasan, C., and Mitkov, R. (2019). Enhancing unsupervised sentence similarity methods with deep contextualised word representations.

In *Proceedings of the International Conference on Recent Advances in Natural Language Processing*.

Sansonetti, G., Gasparetti, F., Micarelli, A., Cena, F., and Gena, C. (2019). Enhancing cultural recommendations through social and linked open data. *User Modeling and User-Adapted Interaction*, 29(1):121–159.

Srivastava, S. and Singh, A. K. (2018). Graph based analysis of panama papers. In *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, pages 822–827. IEEE.

Stalnaker, R. (2002). Common ground. *Linguistics and philosophy*, 25(5/6):701–721.

Stray, J. (2019). Making artificial intelligence work for investigative journalism. *Digital Journalism*, 7(8):1076–1097.

Van der Maaten, L. and Hinton, G. (2008). Visualizing data using t-sne. *Journal of machine learning research*, 9(11).

Webber, J. (2012). A programmatic introduction to neo4j. In *Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity*, pages 217–218. ACM.

Yang, L., Xiao, Z., Jiang, W., Wei, Y., Hu, Y., and Wang, H. (2020). Dynamic heterogeneous graph embedding using hierarchical attentions. *Advances in Information Retrieval*, 12036:425.

Yao, L., Zhang, Y., Wei, B., Jin, Z., Zhang, R., Zhang, Y., and Chen, Q. (2017). Incorporating knowledge graph embeddings into topic modeling. In *Thirty-first AAAI conference on artificial intelligence*.

Yu, H., Li, H., Mao, D., and Cai, Q. (2020). A domain knowledge graph construction method based on wikipedia. *Journal of Information Science*, page 0165551520932510.

Zhou, Y., Zhang, L., and Yi, Z. (2019). Predicting movie box-office revenues using deep neural networks. *Neural Computing and Applications*, 31(6):1855–1865.