Information Retrieval (IR)
Project

Project Title:
Knowledge Graph-Based on Movies.

Group Id: 5

Group Members :
Anirudh Jakhotia - S20190010007 - anirudh.j19@iiits.in
Harish Mullagura - S20190010124 - harish.m19@iiits.in
Neeraj Dusa - S20190010047 - neeraj.d19@iiits.in
V Venkata Kalyan - S20190010193 -venkatakalyan.v19@iiits.in

Google Colab Notebook Code Link :
https://colab.research.google.com/drive/1MVY538WLcRjCzgGO_mpvwcpYhwngwy2s?usp=sharing

Dataset Link:
https://drive.google.com/drive/folders/1IpCO8C6V2puZLx5CTgy3HAV7Zps6nF6F?usp=sharing

Problem Statement :
Information Extraction is a process of extracting information in a more structured way i.e., the information which is machine-understandable. It consists of sub-fields that cannot

be easily solved. Therefore, an approach to store data in a structured manner is required for easier access.

Problem Solution:

- ❖ Knowledge Graph is a structured way in which a set of three-item sets called Triple is used. Here, the set combines a subject, a predicate, and an object.

- ❖ We will build a knowledge graph using Python and Spacy. Knowledge graphs put data in context via linking and semantic metadata and this way provide a framework for data integration, unification, analytics, and sharing.

Project Overview :

An IR project based on Knowledge Graph Structures for movies and actors/actresses analysis.

Project Overall Idea:

- ☐ The knowledge graph represents a collection of interlinked descriptions of entities – objects, events, or concepts. The edges are the connections interfacing these elements to each other.

- ☐ We will extricate these components in an unaided way, i.e., we will utilize the punctuation of the sentences.

- ☐ The primary thought is to experience a sentence and concentrate on the subject and the item as and when they are experienced.
- ☐ First, we need to pass the text to the function. The text will be broken down and placed in each token or word in a category. After we have arrived at the finish of a sentence, we clear up the whitespaces which may have remained, and afterward, we're all set, we have gotten a triple.

- ☐ For example, the statement "IIIT Sri City is categorized as a Tier-1 college" will give a triple focus on the main subject(IIIT Sri City, categorized, Tier-1 college).

Description of tasks and subtasks in the project:

Sentence Segmentation:
The first step in building a knowledge graph is to split the text document or article into sentences. Then, we will shortlist only those sentences in which there is exactly 1 subject and 1 object.

Entities Extraction :
We easily do this with the help of parts of speech (POS) tags. The nouns and the proper nouns would be our entities. However, when an entity spans multiple words, then POS tags alone are not sufficient.

We need to parse the dependency tree of the sentence. For dependency parsing, the rule can be defined as extracting the subject/object along with its modifiers, compound words and also extracting the punctuation marks between them.

Extract Relations :
To build a knowledge graph, we need edges to connect the nodes (entities) to one another. These edges are the relations between a pair of nodes. To extract the relation, we have to find the ROOT of the sentence which is also the verb of the sentence.

Build a Knowledge Graph from Text Data :
We will build a knowledge graph from scratch by using the text from a set of movies and films related to Wikipedia articles. The dataset used here is wiki_sentences_v2.csv

Data Preprocessing :

Entity Pairs Extraction:
The nodes are going to be the entities that are present in the Wikipedia sentences. Edges are the relationships connecting these entities to one another.

We will extract these elements in an unsupervised manner, i.e., we will use the grammar of the sentences. The main idea

is to go through a sentence and extract the subject and the object as and when they are encountered.

Predicate Extraction:
The pattern defined in the function tries to find the ROOT word or the main verb in the sentence. Once the ROOT is identified, then the pattern checks whether it is followed by a preposition (‘prep’) or an agent word. If yes, then it is added to the ROOT word.

```
[ ]  def get_relation(sent):

        doc = nlp(sent)

        # Matcher class object
        matcher = Matcher(nlp.vocab)

        #define the pattern
        pattern = [{'DEP':'ROOT'},
                   {'DEP':'prep','OP':"?"},
                   {'DEP':'agent','OP':"?"},
                   {'POS':'ADJ','OP':"?"}]

        matcher.add("matching_1", None, pattern)

        matches = matcher(doc)
        k = len(matches) - 1

        span = doc[matches[k][1]:matches[k][2]]

        return(span.text)
```

Let’s take a look at the most frequent relations or predicates that we have just extracted:

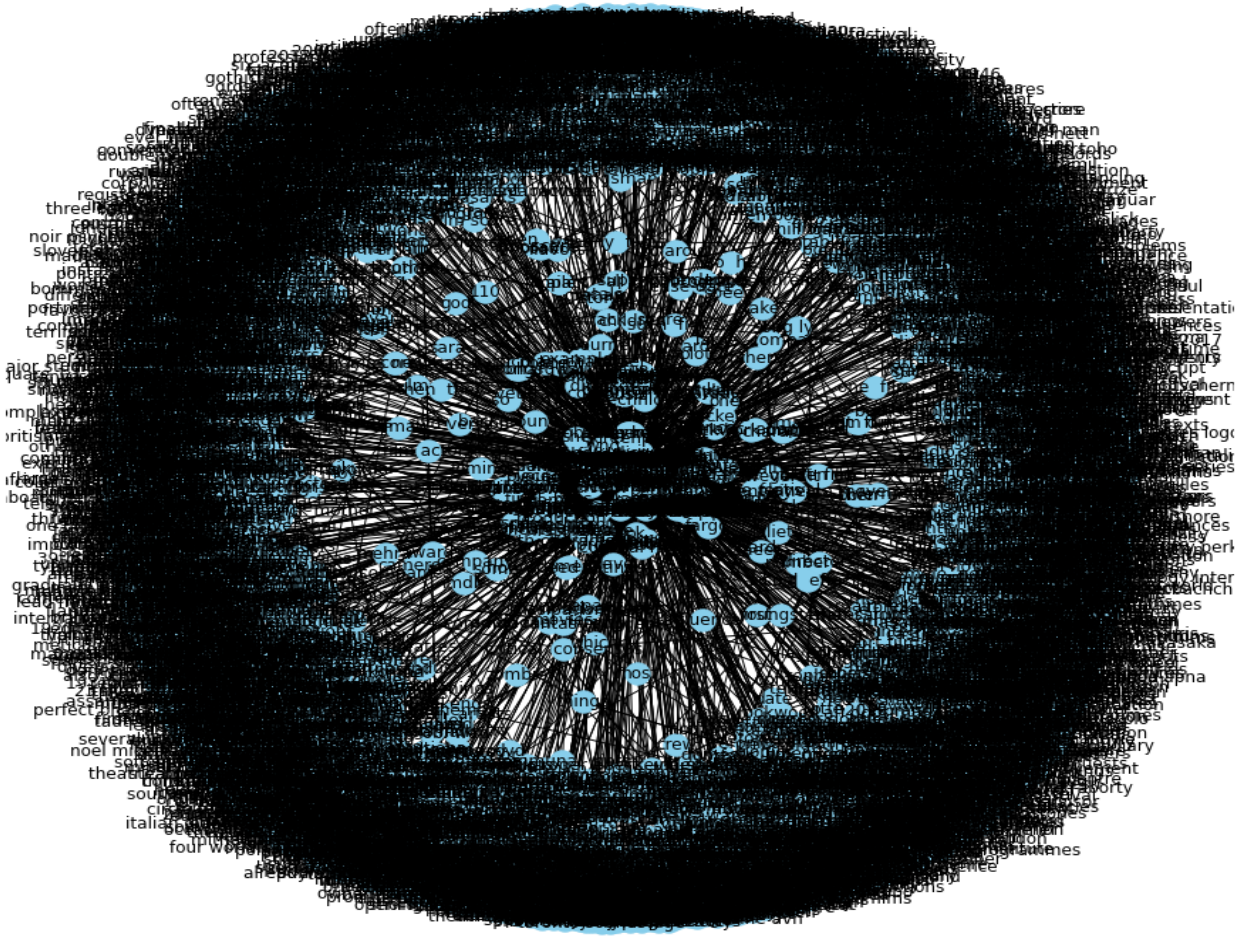
```
[ ]  pd.Series(relations).value_counts()[:50]

is                370
was               297
released on        87
include            73
were               71
are                71
released           40
's                 38
composed by        35
has                31
have               31
became             31
become             29
released in        27
included           26
produced           22
called             22
```

Building a Knowledge Graph : (Indexing and Evaluation)

We will finally create a knowledge graph from the extracted entities (subject-object pairs) and the predicates (relation between entities).

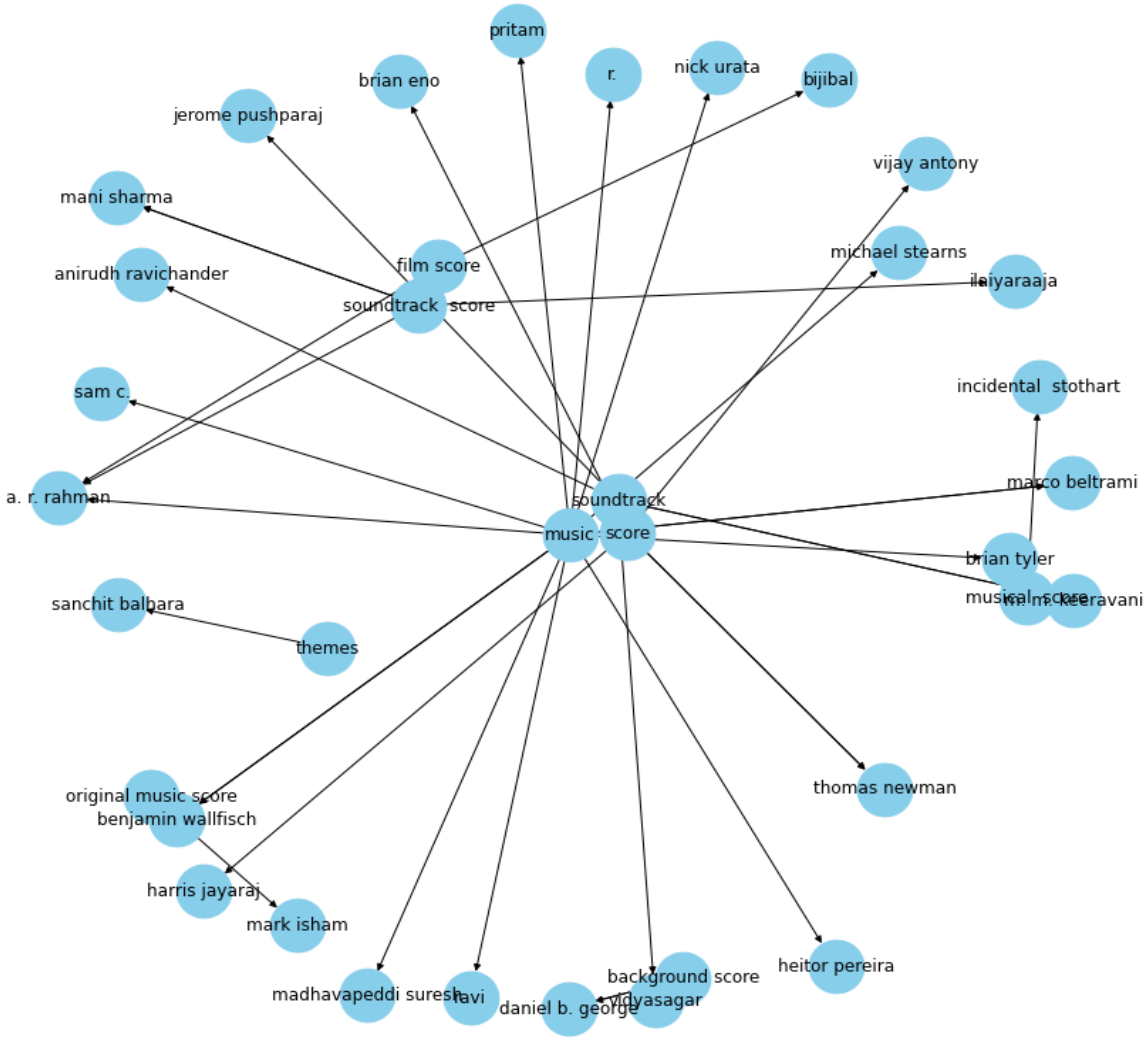
```
[ ]  # extract subject
     source = [i[0] for i in entity_pairs]

     # extract object
     target = [i[1] for i in entity_pairs]

     kg_df = pd.DataFrame({'source':source, 'target':target, 'edge':relations})
```

Next, we will use the *networkx* library to create a network from this data frame. The nodes will represent the entities and the edges or connections between the nodes will represent the relations between the nodes. It is a directed graph.



It becomes really hard to visualize a graph with these many relations or predicates.

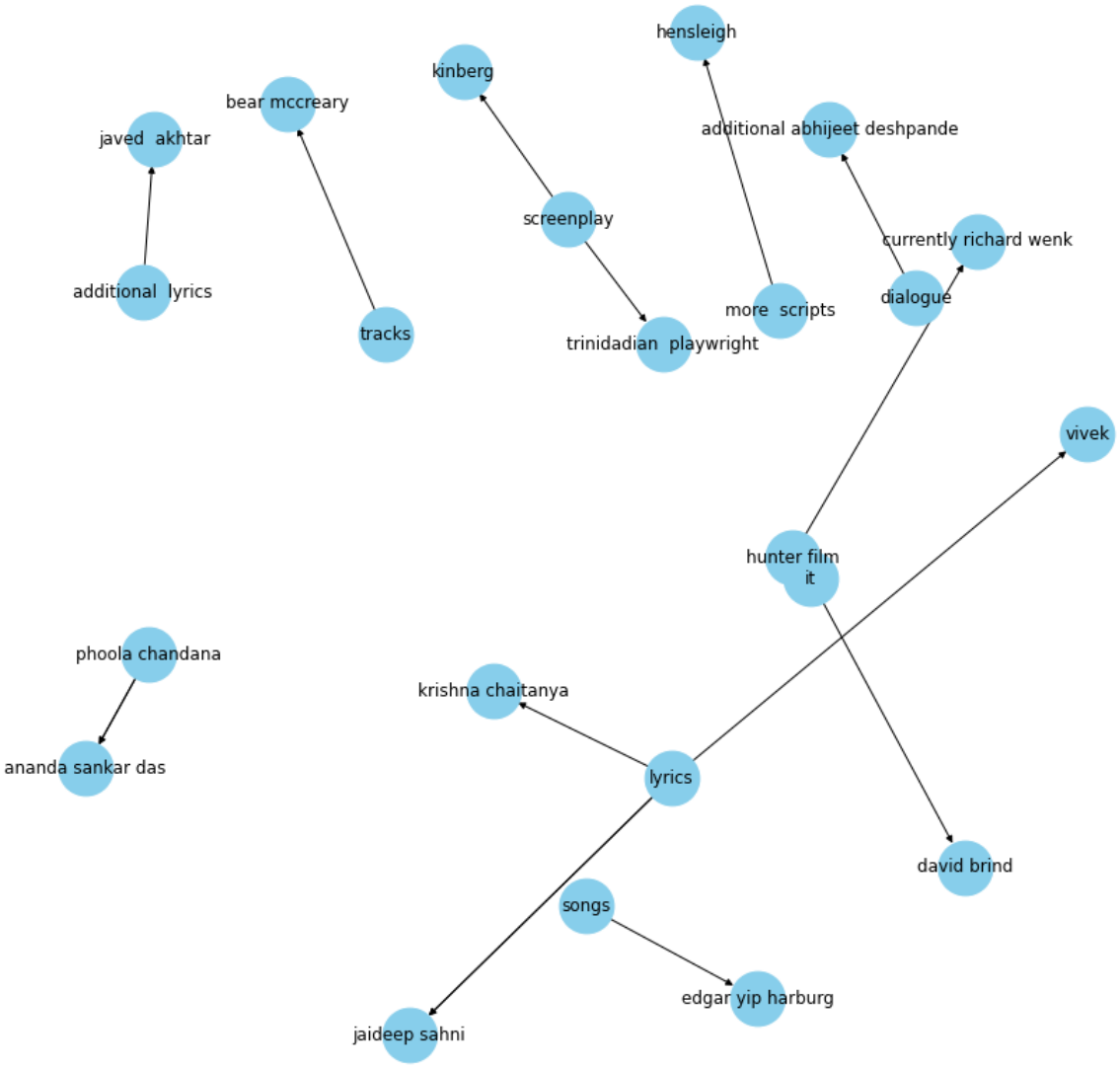
Knowledge Graph for relation “composed by”:



That’s a much cleaner graph. Here the arrows point towards the composers. For instance, A.R. Rahman, who is a renowned music composer, has entities like “soundtrack score”, “film score”, and “music” connected to him in the graph above.

Knowledge Graph for relation “written by”:

Since writing is an important role in any movie, I would like to visualize the graph for the “written by” relation.

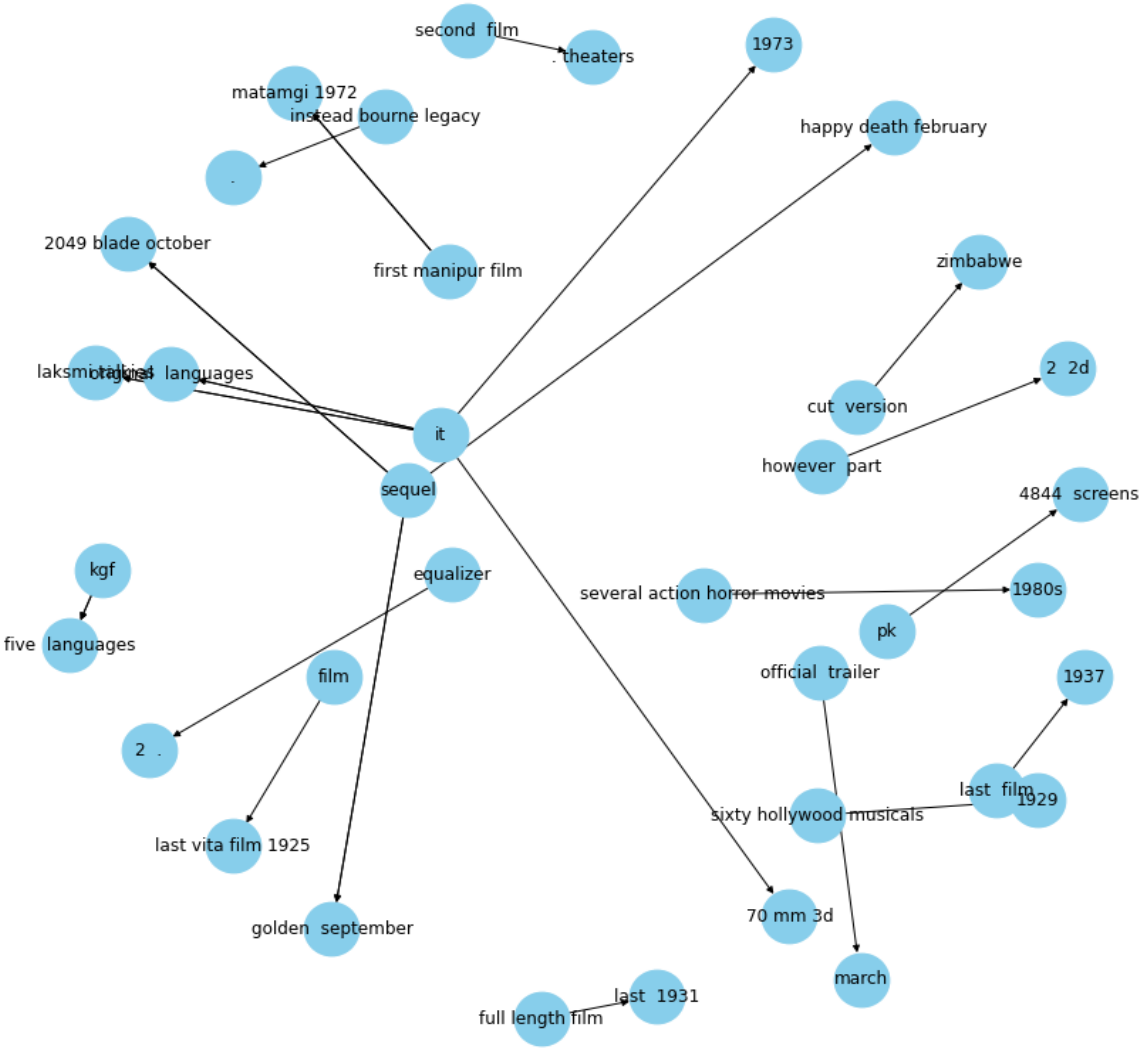


This knowledge graph is giving us some extraordinary information. Guys like Javed Akhtar, Krishna Chaitanya, and

Jaideep Sahni are all famous lyricists and this graph beautifully captures this relationship.

Let’s see the knowledge graph of another important predicate, i.e., the “released in”:

Knowledge Graph for relation “released in”:



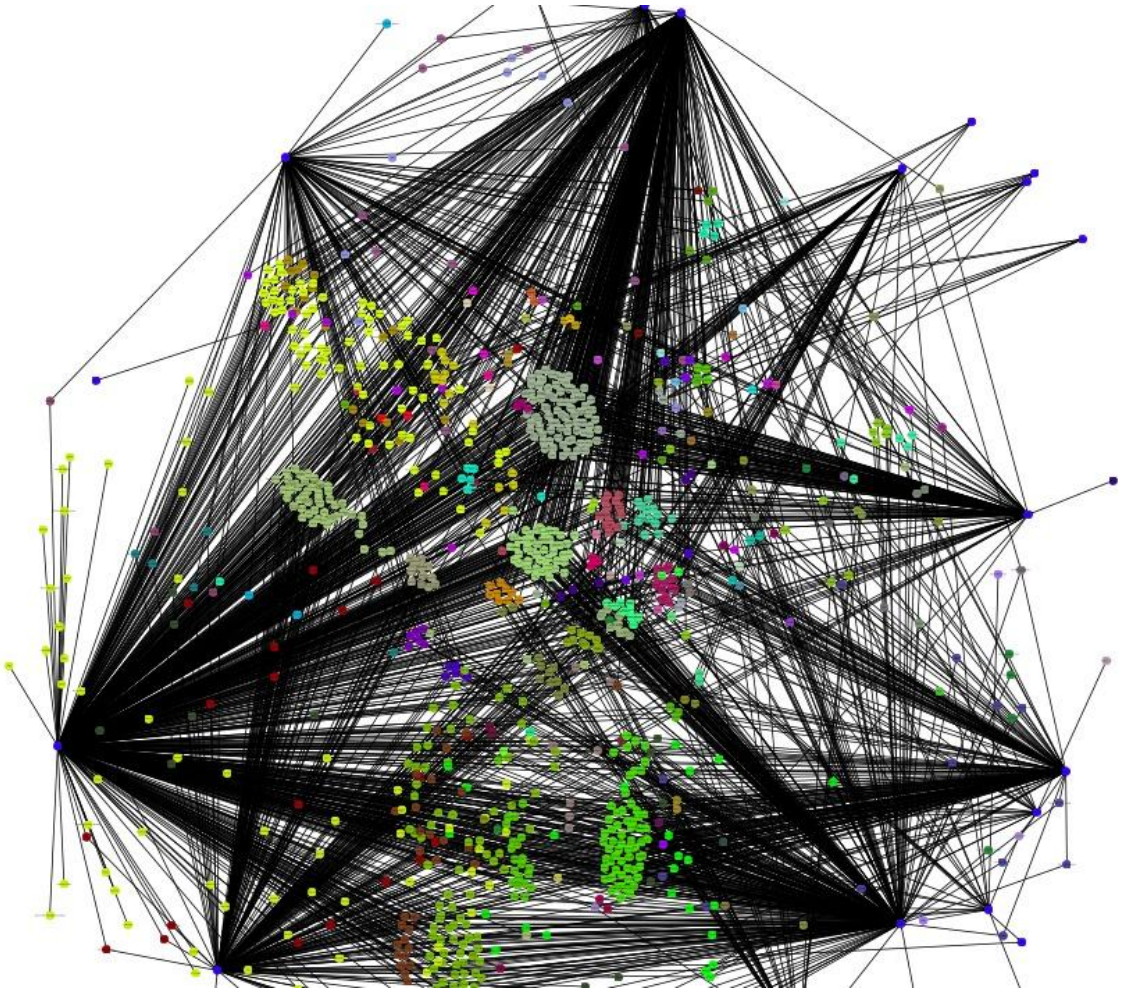
“There are several action horror movies released in the 1980s” and “pk released on 4844 screens”. These are facts and it shows us that we can mine such facts from just text.

Ranked Retrieval model :(Indexing And Ranking)

```
[ ]  get_movie_recommendation('Memento')
```

	Title	Cosign Distance
1	American Beauty (1999)	0.389346
2	American History X (1998)	0.388615
3	Pulp Fiction (1994)	0.386235
4	Lord of the Rings: The Return of the King, The...	0.371622
5	Kill Bill: Vol. 1 (2003)	0.350167
6	Lord of the Rings: The Two Towers, The (2002)	0.348358
7	Eternal Sunshine of the Spotless Mind (2004)	0.346196
8	Matrix, The (1999)	0.326215
9	Lord of the Rings: The Fellowship of the Ring,...	0.316777
10	Fight Club (1999)	0.272380

Knowledge graph of 1000 movies:



The blue nodes near the edges of the graph represent genres, edges of the same color represent the same genre movies.

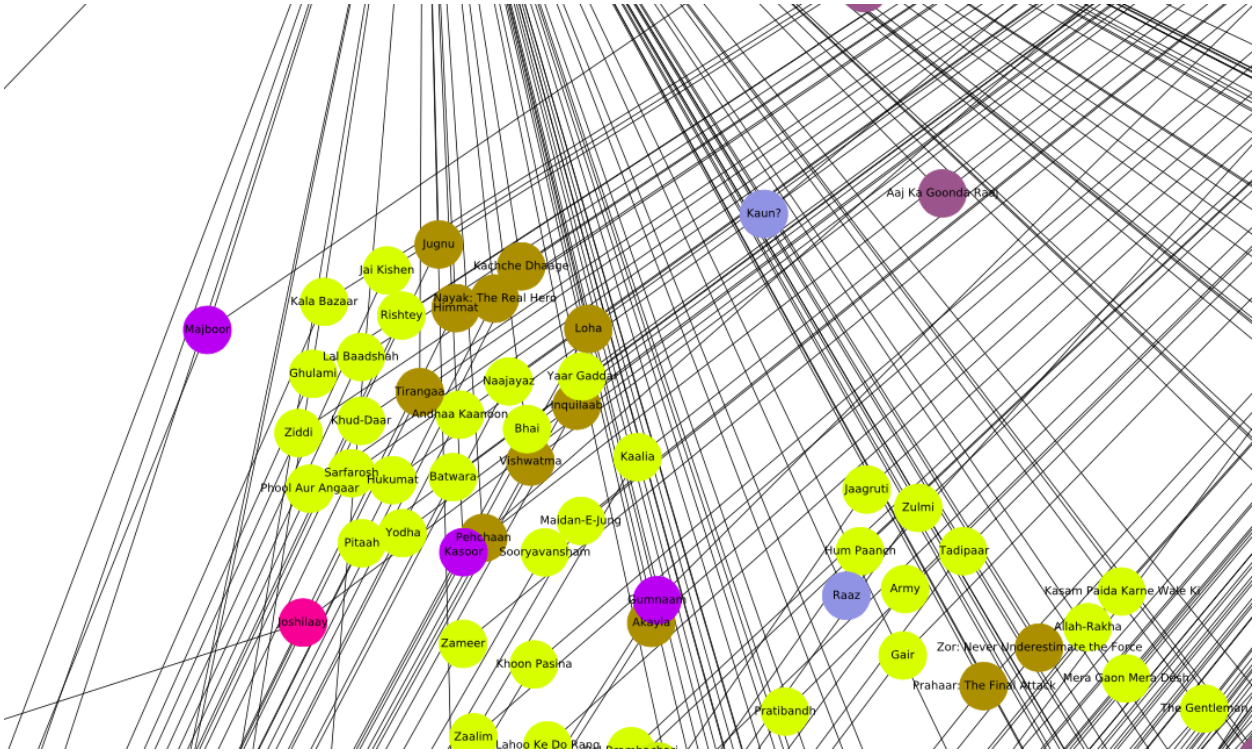
A movie node's color is derived by the combination of its genre’s color. Therefore movies with similar characteristics would be of the same color and in the same color.

Properties of graph :

a. There is homogeneity between the movies belonging to the same color. (Represented by the same color.)

b. There is heterogeneity among movies from different clusters. (Represented by different colors.)

Zoomed pic of a cluster:



Here you can see the movies which belong to different colors. Each color represents a color.

Such a graph provides us insight into what movie a user can like based on a few of his judgments.

Team Contributions:

Anirudh Jakhotia - Movie detail graph for a song knowledge graph, Spell Correction for the entertainment domain, Ranking

Harish Mullagura - Knowledge Graph for Movies, Writer knowledge graph, entity recognition, and classification, Ordering

Neeraj Dusa - Movies released knowledge graph, Knowledge graph of 1000 movies, Entity recognition, and classification, Indexing

V Venkata Kalyan - Movies Genre Graph, Evaluation.

Conclusion

We extracted information from a given text in the form of triples and build a knowledge graph from it. However, we restricted ourselves to using sentences with exactly 2 entities. Even then we were able to build quite informative knowledge graphs.

We have described how KGs can help IR by discussing several entity-centric IR tasks and the role of KGs in each. Specifically, we discussed entity linking, document retrieval, entity retrieval, entity recommendation, and relationship explanation.

Thus, we were able to build a Knowledge graph for the movies dataset.